# DISCOVERING AND EXPLOITING STRUCTURE FOR GAUSSIAN PROCESSES

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Jacob Ross Gardner

May 2018

DISCOVERING AND EXPLOITING STRUCTURE FOR GAUSSIAN PROCESSES

Jacob Ross Gardner, Ph.D.

Cornell University 2018

Gaussian processes have emerged as a powerful tool for modeling complex and noisy functions. They have found wide applicability in personalized medicine, time series analysis, prediction tasks in the physical sciences, and recently black-box optimization. Their success is in large part due to two fundamental advantages they enjoy over many other models. First, they provide convenient and often well-calibrated uncertainty estimates. Machine learning models make mistakes, and by offering users full probabilistic predictions, Gaussian processes allow for more informed decision making in the face of uncertainty. Second, they allow users to encode and incorporate prior knowledge about their modelling task through the use of flexible and composable *covariance functions* or *kernels*. This aspect of Gaussian processes is particularly critical when faced with functions that are expensive or burdensome to evaluate, as they allow users to develop models that will generalize even with very limited data–in some cases, even before the first training example is collected.

Despite these two clear advantages, some of the most popular applications of Gaussian processes have focused on exploiting the first advantage of GPs, and very little on exploiting the latter. As an example, in Bayesian optimization, off-the-shelf implementations often use the most generic and flexible covariance functions available. While *a priori* this ensures the generality of Bayesian optimization, it can significantly increase the number of function evaluations requred

to perform optimization.

In this thesis, we will demonstrate by way of application that the second advantage can be just as critical as the first. By leveraging expert medical knowledge, we develop a GP model that exploits basic facts about human hearing to dramatically improve both the quality and speed of modern audiometric testing. By automatically discovering independence structure in an objective function, we can leverage recent work on additive structure in Bayesian optimization to achieve *exponentially* lower sample complexities. Finally, by exploiting the product structure inherent to the RBF kernel–arguably the most common covariance function in usage–we will develop approximations for the GP marginal log likelihood that result in *exponential* improvement to the running time complexity of Gaussian process inference compared to existing inducing point methods, resulting in the fastest asymptotic time complexity for training we are aware of.

For my wife and best friend, Katie. I could not function without your love, support, and unbounded patience.

P.S. Have you seen my wallet?

## BIOGRAPHICAL SKETCH

Jacob Gardner was born in 1990 in Raleigh, North Carolina where he attended Ravenscroft for kindergarten through highschool. He received his B.S. in 2013 from Washington University in St. Louis in an independently designed major focusing on computational biology. He started his Ph.D. in Computer Science at Washington University in 2013 advised by Kilian Weinberger, before moving with the lab after to Cornell University in 2015. After graduating, he will be starting as a postdoctoral associate in Operations Research and Information Engineering.

## ACKNOWLEDGEMENTS

The work I have done over the last five years would not have been possibly without the guidance and help of so many amazing and talented people along the way. I would like to thank my advisor, Kilian Weinberger. I've had a lot of people teach me a lot of things in graduate school, but Kilian taught me what you are truly supposed to learn in graduate school: how to be a scientist. He taught me how to design experiments that validate a claim or truly get to the bottom of a problem; the importance of always trying the simple solution first; and the value of clear and effective communication. I would also like to thank my other committee members, Kavita Bala and Karthik Sridharan, for their always helpful comments, insight, and dedication to my success as a graduate student.

This thesis deals with Gaussian processes and Bayesian machine learning. These are tools that I found to be conceptually very difficult to grasp when I started my degree. Fortunately, I have had the pleasure of working with many exceptional experts in the area over the past five years, including John Cunningham, Roman Garnett, and Andrew Wilson. I would like to thank them for their guidance on everything Bayesian. The last chapter of this thesis and my recent work since has had a large focus on numerical linear algebra techniques, and I'd like to thank David Bindel for an excellent class and useful discussions in this area.

Finally, I'd like to thank the outstanding labmates I've had throughout the years. Thanks to Stephen Tyree and Eddie Xu for being the realist foils to my usually excessive optimisim; to Matt Kusner for being a paragon of work ethic; to Geoff Pleiss for his guidance in software development; to Chuan Guo for always having the principled point of view; and to Felix, for finding the simplest solution for any problem.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

CHAPTER 1

# INTRODUCTION

Gaussian processes (GPs) have emerged as a powerful probabilistic tool for modeling complex and noisy functions. They have found wide applicability in personalized medicine, time series analysis, prediction tasks in the physical sciences, and recently blackbox optimization. Their success is in large part due to two fundamental advantages they enjoy over many other models. First, they provide convenient and often well-calibrated uncertainty estimates. Machine learning models make mistakes, and by offering users full probabilistic predictions, Gaussian processes allow for more informed decision making in the face of uncertainty. Second, they allow users to encode and incorporate prior knowledge about their modelling task through the use of flexible and composable *covariance functions* or *kernels*. This aspect of Gaussian processes is particularly critical when faced with functions that are expensive or burdensome to evaluate, as they allow users to develop models that will generalize even with very limited data–in some cases, even before the first training example is collected.

This thesis focuses on this second aspect of Gaussian processes: taking advantage of the structure imposed on the model by the choice of covariance function. In particular, we seek to answer two fundamental questions (1) how can we intelligently select a prior (i.e., covariance function) *even when faced with little or no data at all?*, and (2) once a prior has been selected, how do we best exploit the structure encoded in that prior? In the next two section, we will make these questions more concrete and provide motivation for them.

## 1.1 Discovering covariance structure

The ability to encode prior knowledge about a function through the covariance function is one of the defining aspects and key advantages of the Gaussian process. When strong domain knowledge about a modeling task is available, selecting an appropriate covariance function can allow GPs to accurately model a function with very little data. This fact is arguably at the heart of the GP's success in the low data regime.

In Figure 1.1, we illustrate this point on a simple toy regression task for clarity. Six training examples are sampled from a true function, and a Gaussian process is then trained with four different covariance functions. The lower right plot depicts a Gaussian process model with a covariance function that exactly encodes the class of functions that the true function belongs to. As a result, it is able to exactly recover the true function, *despite being limited to only six training examples*.

Choosing a strong covariance function to encode prior knowledge has one major caveat: covariance functions that encode strong prior knowledge are often not flexible or general purpose. They significantly narrow the hypothesis class that can be used to explain the training data. For example, the *periodic kernel* discussed later can *only* give rise to models that exactly repeat. As a result, care must be taken to ensure that the chosen covariance function encodes valid assumptions, or the model's generalization ability will suffer greatly. This can be burdensome on users, and in practice–despite the rich and powerful language of covariance functions available–standard practice in off the shelf Gaussian processes is to use overly generic kernels. This motivates techniques that select

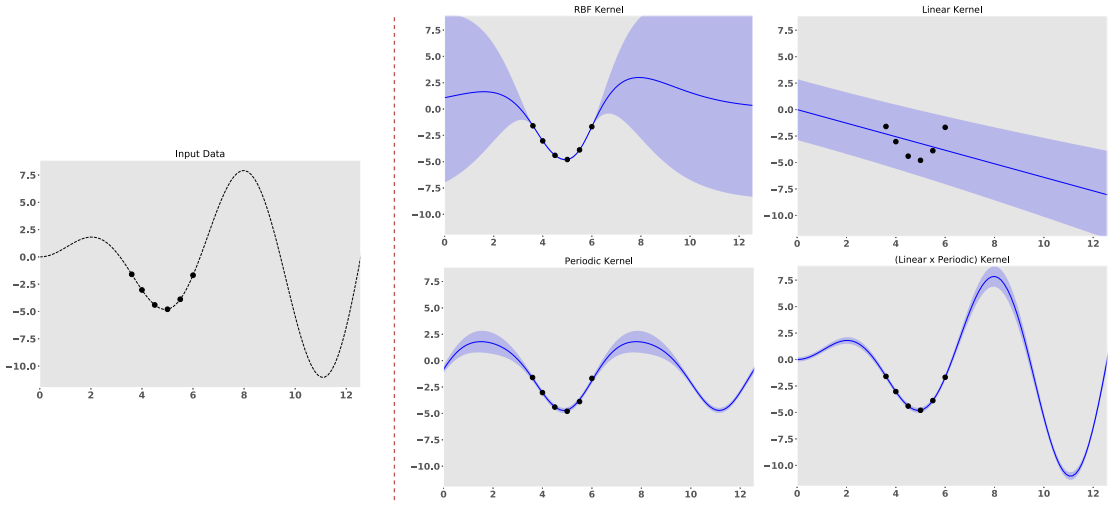


Figure 1.1: A simple toy regression problem **(left)**, along with four possible GP models given the six input datapoints **(right)**. Each model corresponds to a different choice of *covariance function*. "Periodic x Linear" corresponds to a composition of two elementary covariance functions. Composing covariance functions is discussed further in Section 2.2.

covariance functions automatically. Recently proposed methods exist for automatically and effectively discovering covariance structure [19]. However, these techniques typically assume a somewhat large pre-existing set of data, which precludes their application in the limited data setting.

## 1.2 Exploiting covariance structure

The procedures necessary to train Gaussian processes are often prohibitively expensive, and prevent their application to tasks with more than a few thousand data points or a handful of input dimensions without significant losses in performance. Traditional techniques for training Gaussian processes involve computing linear solves and log determinants with an $n \times n$ positive definite and symmetric *kernel matrix* $K$–operations that require $\mathcal{O}(n^3)$ time. Recent work on so-called *inducing point methods* construct a small set of $m < n$ inducing points

that allow for the construction of a low-rank approximation of $K$. This allows inference to be performed in $\mathcal{O}\left(nm^2\right)$ time [97, 89].

Structured kernel interpolation (SKI) [113] is a recently proposed inducing point method that, given a regular grid of $m$ inducing points, allows for inference to be performed in $\mathcal{O}\left(n + m \log m\right)$ time–an impressive improvement over other inducing point methods. Unfortunately, to achieve this impressive running time, the number of inducing points $m$ grows exponentially with the dimensionality of the inputs, limiting the applicability of SKI to problems with fewer than $5$ input dimensions. In this thesis, we will demonstrate that by exploiting product structure displayed by many popular covariance functions, we can reduce this exponential dependence on input dimensionality to *linear time*.

## 1.3 Outline

This thesis details the following contributions for dealing with these two questions:

1. In Chapter 2, we will describe the basic mathematical background that this thesis builds upon. We give an overview of Gaussian processes, including covariance function composition and model selection. We review Bayesian optimization and Bayseian active learning, two key applications that will appear later in the thesis. We additionally review techniques for scalable Gaussian process inference, including recent work on inducing point methods.

2. In Chapter 3, we develop a novel audiometric detection (hearing) test based on *active learning* with Gaussian processes. This is a real-world case study

motivating the need for intelligent covariance function selection in the small data regime. By choosing a likelihood and covariance function using expert domain knowledge, we are able to develop a hearing test that is significantly faster and more accurate than state-of-the-art, yielding accurate test results typically with fewer than 20 tones presented to a patient.

3. In Chapter 4, we extend the active learning techniques from Chapter 2 to not only learn the patient's hearing response function, but also *actively select an appropriate covariance function.* The idea of active learning for model selection will provide us with a tool to quickly choose an appropriate prior for a task in the regime where data collection is expensive.

4. In Chapter 5, we leverage an efficient MCMC sampling strategy to scale the active model selection technique discussed above to deal with choosing from large classes of priors. We apply this to the problem of Bayesian optimization for high dimensional functions, and develop a technique that–by automatically discovering whether a function decomposes additively and exploiting this structure–is able to significantly outperform standard Bayesian optimization techniques when faced with more than three or four input dimensions.

5. In Chapter 6, we describe how to leverage covariance function structure for efficient inference. We describe a technique based on structured kernel interpolation (SKI) [113] that can exploit structure existing in certain types of covariance function compositions. This technique results in an *exponential improvement* in the running time complexity of SKI when faced with high dimensional data.

CHAPTER 2

# MATHEMATICAL BACKGROUND

## 2.1 Gaussian processes

A Gaussian process generalizes the multivariate normal distribution to define a distribution over functions. Analogous to the mean vector and covariance matrix of the multivariate normal distribution, a GP is specified by a *prior mean function* and *prior covariance function* $f(\mathbf{x}) \sim \mathcal{GP}\left(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right)$. Gaussian processes by definition have the property that the function values at any finite set of inputs $[\mathbf{x}_1, ..., \mathbf{x}_N]$ are jointly Gaussian distributed:

$$\mathbf{f} = [f(\mathbf{x}_1), ..., f(\mathbf{x}_N)] \sim \mathcal{N}\left(\mu_X, K_{XX}\right)$$

where $\mu_X = [\mu(\mathbf{x}_1), ..., \mu(\mathbf{x}_N)]^\top$ and $K_{XX} = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^N$. Throughout, we will denote by $K_{AB}$ the kernel matrix achieved by applying the kernel function to all pairs of points with one from $A$ and one from $B$, $(\mathbf{a}_i, \mathbf{b}_j)$. While $\mu(\cdot)$ can be chosen to be essentially any function, the covariance function has a typical restriction: in the case where $A = B$, $K$ must be a valid covariance matrix (i.e., it must be symmetric positive definite). For practical purposes, this restriction is often relaxed–some covariance functions may result in $K$ having some zero eigenvalues.

It is common in Gaussian process regression to assume an additive noise model relating function values to observed measurements. That is, we assume an observation $y_i$ is related to the observed features $\mathbf{x}_i$ according to $y_i = f(\mathbf{x}_i) + \epsilon$, where $\epsilon \sim \mathcal{N}\left(0, \sigma^2\right)$. Given a training dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$, the goal of inference is to infer the posterior distribution over the latent function, $p(f \mid \mathcal{D})$.

Define $\hat{K}_{XX} = [K_{XX} + \sigma^2 \mathbf{I}]$ for notational simplicity. Under the Gaussian noise observation model this distribution is again a Gaussian process:

$$p(f(\mathbf{x}^*) \mid \mathcal{D}) \sim \mathcal{GP}\left(\mu_{f|\mathcal{D}}(\mathbf{x}^*), k_{f|\mathcal{D}}(\mathbf{x}^*, \mathbf{x}^{*'})\right),$$

$$\mu_{f|\mathcal{D}}(\mathbf{x}) = \mu(\mathbf{x}^*) + K_{\mathbf{x}^* X} \hat{K}_{XX}^{-1} \mathbf{y}, \tag{2.1}$$

$$k_{f|\mathcal{D}}(\mathbf{x}^*, \mathbf{x}^*) = K_{\mathbf{x}^*\mathbf{x}^*} - K_{\mathbf{x}^* X} \hat{K}_{XX}^{-1} K_{\mathbf{x}^* X}^\top, \tag{2.2}$$

These equations can be derived using standard Gaussian conditioning and sum rules. This follows directly from the definition of the Gaussian process above: the *observed training labels* $y_1, ..., y_n$ are the (noisy) observed function values for $\mathbf{x}_1, ..., \mathbf{x}_n$. Given some set of test points $X^*$, the definition of a GP immediately implies that $f(X^*) = [f(\mathbf{x}_1^*), ..., f(\mathbf{x}_t^*)]$ and $[y_1, ..., y_n]$ are jointly Gaussian distributed:

$$\begin{bmatrix} \mathbf{y} \\ f(X^*) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{y}) \\ \mu(f(X^*)) \end{bmatrix}, \begin{bmatrix} K_{XX} + \sigma^2 \mathbf{I} & K_{XX^*} \\ K_{XX^*}^\top & K_{X^*X^*} \end{bmatrix}\right). \tag{2.3}$$

This allows us to immediately derive equations (2.1) and (2.2) using standard properties of Gaussian processes.

### 2.1.1 Hyperparameter learning

All kernel matrices implicitly depend on hyperparameters $\theta$. The log *marginal likelihood* of the data, conditioned only on these hyperparameters, is given by

$$\log p(\mathbf{y} \mid \theta) = -\frac{1}{2}\mathbf{y}^\top \hat{K}_{XX}^{-1}\mathbf{y} - \frac{1}{2}\log|\hat{K}_{XX}| + \mathrm{c}\,, \tag{2.4}$$

All prior hyperparameters can be treated automatically, either by maximizing or by choosing a prior $p(\theta)$ and sampling from $p(\theta \mid \mathcal{D}) \propto p(\mathbf{y}|\theta)p(\theta)$.

## 2.2 Covariance functions

As discussed above, the two entities that must be specified to describe a Gaussian process prior are the *prior mean function* and *prior covariance function*. In principle, any function $k(\mathbf{x}, \mathbf{x}')$ that results in $K_{XX}$ being positive definite and symmetric is a valid choice of covariance function. In machine learning in particular, such functions have often been studied in the context of *kernel methods* (e.g., [88]), and the functions themselves are typically referred to as *kernels*.

In the context of Gaussian processes, kernels specify the covariance (and hence, roughly the degree of correlation) between the function values of two inputs, $\mathbf{x}$ and $\mathbf{x}'$. Beyond this, kernels specify what functions are likely *a priori*.

There are in some sense two "views" of how kernels can best be chosen and utilized depending on how much data is available.

**Composing simple kernels.** The first option is to choose or compose relatively simple commonly used kernels.

There are a variety of ways in which kernels can be combined to perform new kernels, but in this thesis we will primarily focus on two: addition and multiplication. In particular, if $k_a(\mathbf{x}, \mathbf{x}')$ and $k_b(\mathbf{x}, \mathbf{x}')$ are kernels, then the following are also kernels:

$$k_{a+b} = k_a(\mathbf{x}, \mathbf{x}') + k_b(\mathbf{x}, \mathbf{x}') \tag{2.5}$$

$$k_{a\times b} = k_a(\mathbf{x}, \mathbf{x}') \times k_b(\mathbf{x}, \mathbf{x}') \tag{2.6}$$

This follows immediately from the fact that the set of positive definite matrices is closed under addition and elementwise multiplication. In particular, letting

$A \circ B$ denote the elementwise matrix product, the training covariance matrices resulting from the above kernels,

$$K_{a+b} = K_a + K_b \tag{2.7}$$

$$K_{a \circ b} = K_a \circ K_b \tag{2.8}$$

are both positive definite.

This strategy often allows practioners to encode prior knowledge about their modeling function in their choice of Gaussian process prior, which can allow for learning even in the presence of very little data. As an example, in Figure 1.1, data for a simple toy regression problem and the true underlying function is plotted, as well as four possible Gaussian process models. Each model corresponds to a different covariance function–three very common choices of covariance function, as well as a composition of two covariance functions. The model resulting from a product of a linear kernel and a periodic kernel results in a very accurate model of the underlying function, even though only six data points are available.

## 2.3 Bayesian optimization

Gradient-free optimization of expensive black-box functions is a ubiquitous task in a many fields. Applications range from sensor placement [27], to robotic control [10], to hyperparameter tuning for complex machine learning algorithms such as deep convolutional neural networks [90]. These optimization tasks are challenging, because we typically have little knowledge about the objective function beyond the ability to evaluate it at a chosen location, and because the function is expensive to evaluate. For example, each lab experiment costs time and

money, and evaluating a given set of hyperparameters for a deep neural network may take days or even weeks of training time.

Bayesian optimization (BayesOpt) [64, 7] is a widely used technique for optimizing expensive black-box functions. Broadly, BayesOpt operates by maintaining a probabilistic belief about the objective function. We place a prior distribution—most often a Gaussian process—over the objective function, which we condition on data as we observe it. We use this belief to drive the optimization process by sequentially evaluating the objective. At each iteration, the posterior distribution of the function is used to suggest queries to evaluate next, typically trading off exploration (i.e., querying areas with high posterior uncertainty) and exploitation (i.e., querying areas with low posterior mean).

More formally, Bayesian optimization (BayesOpt) is a technique for optimizing an expensive black-box function [64]:

$$\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$$

It has become popular in the machine learning community in recent years for its application to hyperparameter tuning [90], but as a strong global optimization algorithm with many theoretical results [92, 8], it has been successfully applied in many other areas as well [27, 10, 58, 12]. For a comprehensive overview of BayesOpt see [7]; we give a brief overview of the main components below.

The core idea of BayesOpt is to deal with the fact that $f(\mathbf{x})$ is expensive by constructing a model of $f$ given samples seen so far, and then using this model to decide where to sample next. This new sample reveals new information about the function, and our model of it can therefore be updated; the process then repeats. Intuitively, if the model of $f$ is perfect and can be optimized cheaply, then the function itself can be optimized cheaply. However, when we are faced

with very few and noisy samples the model will typically be far from perfect. It is therefore critical that the model have well-calibrated uncertainty estimates.

An implementation of Bayesian optimization requires specifying two entities: the model to use for $f$, and an *acquisition function* that evaluates or estimates how useful a point would be to sample next. In this thesis, we will exclusively consider Gaussian processes as the choice to model $f$, as they naturally provide uncertainty estimates with their predictions, and are by far the most common model employed for this task.

**Acquisition functions.** After conditioning on data, we use the predictive distribution (equations (2.1) and (2.2)) to evaluate how promising each candidate $\mathbf{x}^*$ is using some *acquisition function.* One of the most-popular acquisition functions used in BayesOpt is the *expected improvement*. Let $f_{\text{best}}$ be the best function value observed so far. Define $I(f(\mathbf{x}^*))$ as the *improvement* obtained by sampling at $\mathbf{x}^*$:

$$I\big(f(\mathbf{x}^*)\big) = \max\big(0, f_{\text{best}} - f(\mathbf{x}^*)\big)$$

The EI acquisition function evaluates the expectation of the above function over the predictive posterior

$$\begin{aligned}\text{EI}(\mathbf{x}^*) &= \mathbb{E}\left[I\big(f(\mathbf{x}^*)\big)\right]_{p(f(\mathbf{x}^*)|\mathcal{D},\mathbf{x}^*)} \\ &= \int_{-\infty}^{f_{\text{best}}} \big(f_{\text{best}} - f(\mathbf{x}^*)\big) p(f(\mathbf{x}^*) \mid \mathcal{D}, \mathbf{x}^*)\, \mathrm{d}f(\mathbf{x}^*)\end{aligned}$$

The expected improvement may be evaluated analytically. Further, the form of the expected improvement has two terms that can naturally be interpreted as encouraging both *exploitation*, *i.e.* points that our model suggest have low predictive mean $\mu_{f|\mathcal{D}}(\mathbf{x})$, and *exploration*, *i.e.* points with high posterior variance $\sigma_{f|\mathcal{D}}(\mathbf{x}, \mathbf{x})$. To maximize the acquisition function, it is typically evaluated on a

fine grid of candidate points within a pre-defind hyper-cube. Although the evaluation of a single candidate point is very fast, it is important to emphasize that this search scales exponentially with the number of dimensions and can become prohibitively slow in regimes of $10$ or more dimensions.

### 2.3.1 Bayesian active learning

The goal of Bayesian active learning is to sequentially choose samples so as to accurately model an unknown function $f(\mathbf{x})$ with as few samples as possible on some domain $\mathbf{x} \in \mathcal{X}$. From a Bayesian perspective, we would like for our predictive posterior belief $p(y^*|X, \mathbf{y}, \mathbf{x}^*)$ to match $f(\mathbf{x}^*)$ as *well* as possible and as *confidently* as possible. A natural metric that captures how valuable a single candidate point $(\mathbf{x}_t, y_t)$ is when added to an existing dataset $X, \mathbf{y}$ is the *mutual information*, proposed for this purpose by [40]:

$$I(\mathbf{f}, y_t|\mathbf{x}_t) = H\left[\mathbf{f}|X, \mathbf{y}\right] - \mathbb{E}\left[H[\mathbf{f}|X, \mathbf{y}, y_t]\right]_{p(y_t|X,\mathbf{y},\mathbf{x}_t)} \tag{2.9}$$

This is the expected reduction in differential entropy for the latent function, $H[\mathbf{f}]$ after observing the new point $\mathbf{x}_t$. Much as in the Bayesian optimization setting with expected improvement, Bayesian active learning proceeds by sequentially choosing selecting points $\mathbf{x}_t$ that maximize mutual information:

$$\mathbf{x}_t = \arg\max_{\mathbf{x}} I(\mathbf{f}, y|\mathbf{x}) \tag{2.10}$$

In general, computing Equation 2.9 as presented is intractable. However, [40] present a method for approximating it both efficiently and effectively. A re-derivation of this method will be presented in Chapter 3.

### 2.3.2 Bayesian Model Selection

When using Gaussian process regression to model a function $f$, the choice of covariance function $k$ is critical, as it allows the data scientist to encode information about the structure of $f$. However, doing so manually may require expert knowledge about both the function being modeled as well as about kernel methods in general. Bayesian model selection alleviates this need by providing a mechanism to perform inference over the covariance function from a set of candidate kernels. This technique has become popular in recent years for automatic covariance function selection [19].

Suppose we are given a set of observed data $\mathcal{D} = (X, \mathbf{y})$ and a set of kernel functions to choose from, $\mathcal{M}_1, ..., \mathcal{M}_m$, with corresponding hyperparameters $\theta_i$. The first object of interest in Bayesian model selection is the *model evidence* of a model, $p(y \mid \mathcal{M}_i)$. It represents the likelihood of having drawn the dataset $\mathcal{D}$ from a Gaussian process with kernel $K_{\mathcal{M}_i}$. Computing the model evidence exactly is in general not analytically tractable for Gaussian processes. One option is to use the MLE hyperparameters:

$$\log p(y \mid \mathcal{M}_i) \approx p(y \mid \hat{\theta}_i, \mathcal{M}_i). \tag{2.11}$$

The log marginal likelihood for a model $\mathcal{M}_i$ and hyperparameters $\theta_i$ can be computed analytically:

$$\log p(y \mid \mathcal{M}_i, \theta_i) = -\tfrac{1}{2}\mathbf{y}^\top K_{\mathcal{M}_i}^{-1}\mathbf{y} - \tfrac{1}{2}\log\big((2\pi)^n \, |K_{\mathcal{M}_i}|\big).$$

Because the log marginal likelihood does not penalize model complexity, the *Bayesian information criterion* (BIC) is often used to penalize the number of hyperparameters $|\theta_i|$ and better approximate the model evidence:

$$-\tfrac{1}{2}\text{BIC} = \log p(\mathbf{y} \mid \mathcal{M}_i, \theta_i) - \tfrac{1}{2}|\theta_i| \log(|\mathcal{D}|). \tag{2.12}$$

In our experiments, we found that the difference in log marginal likelihood between "correct" additive models and "incorrect" additive models was so large that the penalty term $-\frac{1}{2}|\theta_i| \log\big(|\mathcal{D}|\big)$ did not have a sigificant impact. Given some prior distribution over the possible kernels, $p(\mathcal{M})$, applying Bayes' theorem to the above, and marginalizing out $\theta_i$, results in the *model posterior*,

$$p(\mathcal{M}_i \mid \mathcal{D}) = \frac{p(\mathbf{y} \mid \mathcal{M}_i)p(\mathcal{M}_i)}{\sum_j p(\mathbf{y} \mid \mathcal{M}_j)p(\mathcal{M}_j)}, \tag{2.13}$$

a probability distribution over the possible models given the data. The model posterior provides us with a direct measure of how well each model in the set explains the data. It allows us to generalize eqs. (2.1) and (2.2) to compute a predictive distribution marginalized over all possible models:

$$\mathbb{E}\left[p(f(\mathbf{x}^*) \mid \mathcal{D}, \mathbf{x}^*, \mathcal{M})\right]_{p(\mathcal{M}|\mathcal{D})} \tag{2.14}$$

$$= \sum_i p(f(\mathbf{x}^*) \mid \mathcal{D}, \mathbf{x}^*, \mathcal{M}_i)p(\mathcal{M}_i \mid \mathcal{D}). \tag{2.15}$$

## 2.4 Inference techniques based on matrix-vector multiplies

In order to compute the predictive mean in (2.1), the predictive covariance in (2.2), and the marginal log likelihood in (2.1.1), we need to perform linear solves (i.e. $[K_{XX} + \sigma^2 I]^{-1}\mathbf{v}$) and log determinants (i.e. $\log|K_{XX} + \sigma^2 I|$).

Traditionally, these operations are achieved using the Cholesky decomposition of $K_{XX}$ [73]. Computing this decomposition requires $\mathcal{O}\left(n^3\right)$ operations and storing the result requires $\mathcal{O}\left(n^2\right)$ space. Given the Cholesky decomposition, linear solves can be computed in $\mathcal{O}\left(n^2\right)$ time and log determinants in $\mathcal{O}\left(n\right)$ time.

There exist alternative approaches [113] that require only matrix-vector multiplies (MVMs) with $[K_{XX} + \sigma^2 I]$. To compute linear solves, one can use use the

method of *conjugate gradients* (CG). This technique exploits that the solution to $A\mathbf{x} = \mathbf{b}$ is the unique minimizer of the quadratic function $\frac{1}{2}\mathbf{x}^\top A\mathbf{x} - \mathbf{x}^\top\mathbf{b}$, which can be found by iterating a simple three term recurrence. Each iteration requires a single MVM with the matrix $A$ [85]. Letting $\mu(A)$ denote the time complexity of an MVM with $A$, $k$ iterations of CG requires $O(k\mu(A))$ time. If $A$ is $n \times n$, then CG is exact when $k = n$. However, the linear solve can often be approximated by $k < n$ iterations, since the magnitude of the residual $\mathbf{r} = A\mathbf{x} - \mathbf{b}$ often decays exponentially. In practice the value of $k$ required for convergence to high precision is a small constant that depends on the conditioning of $A$ rather than $n$ [28]. A similar technique known as *stochastic Lanczos quadrature* exists for approximating log determinants in $O(k\mu(A))$ time [17, 99]. In short, inference and learning for GP regression can be done in $\mathcal{O}\left(k\mu(K_{XX})\right)$ time using these iterative approaches.

Critically, if the kernel matrices admit fast MVMs – either through the structure of the data [77, 14] or the structure of a general purpose kernel approximation [113] – this iterative approach offers massive scalability gains over conventional Cholesky-based methods.

## 2.5 Structured kernel interpolation

Structured kernel interpolation (SKI) [113] replaces a user-specified kernel $k(\mathbf{x}, \mathbf{x}')$ with an approximate kernel that affords very fast matrix-vector multiplies. Assume we are given a set of $m$ *inducing points* $U$ that we will use to approximate kernel values.

Instead of computing kernel values between data points directly, SKI com-

putes kernel values between inducing points and *interpolates* these kernel values to approximate the true data kernel values. This leads to the approximate SKI kernel:

$$k(\mathbf{x}, \mathbf{z}) \approx \mathbf{w}_{\mathbf{x}} K_{UU} \mathbf{w}_{\mathbf{z}}^{\top}, \tag{2.16}$$

where $\mathbf{w}_{\mathbf{x}}$ is a sparse vector that contains interpolation weights. For example, when using local cubic interpolation [46], $\mathbf{w}_x$ contains four nonzero elements. Applying this approximation for all data points in the training set, we see that:

$$K_{XX} \approx W_X K_{UU} W_X^{\top} \tag{2.17}$$

With arbitrary inducing points $U$, matrix-vector multiplies with $[W K_{UU} W^{\top}]\mathbf{v}$ require $\mathcal{O}\left(n + m^2\right)$ time. In one dimension, we can reduce this running time by instead choosing $U$ to be a regular grid, which results in $K_{UU}$ being *Toeplitz*. In higher dimensions, a multi-dimensional grid results in $K_{UU}$ being the Kronecker product of Toeplitz matrices. This results in the ability to perform matrix-vector multiplies in at most $\mathcal{O}\left(n + m \log m\right)$ time, and $\mathcal{O}\left(n + m\right)$ storage.

CHAPTER 3

# PSYCHOPHYSICAL DETECTION TESTING WITH BAYESIAN ACTIVE LEARNING

## 3.1 Introduction

In this chapter, we demonstrate the effectiveness of exploiting problem structure when it exists using the real-world task of psychophysical detection testing. We develop a new model for psychophysical detection, guided by experts in the field. While this chapter deals primarily with the technical details and modelling choices made, we note that the primary techniques and results discussed have been thoroughly verified clinically as well [91]. The work in this chapter was published in UAI 2015 [25].

In a psychophysical detection test, a test operator presents $n$ sensory stimuli to a subject, and ask for $n$ binary reports as to whether each stimulus was detected or not. Detection tests exist for vision [79], pain [13], and many other settings. Perhaps the most common example is audiometry [11, 16, 41]. In the standard audiometric detection test, a subject is presented with a sequence of $n$ tones $\mathbf{x}_t \ \ \forall t = 1, \ldots, n$, where each tone $\mathbf{x}_t \in \mathbb{R}^2$ is a pure tone with a specific frequency (pitch) and intensity (volume). The subject reports an observation $y_t = 1$ if they heard the tone (for example, by raising their hand), and a $y_t = 0$ is concluded in the absence of a positive report. The purpose of the test is to infer, from this sequence of observations, the underlying *audiometric function* $g(\mathbf{x})$. An audiometric function is unique to a subject, and describes how likely the subject is to hear sounds over the domain of typical frequencies and intensities. There is substantial variability in each person's audiogram, particularly for those with

partial, selective, or degenerative hearing loss [30, 75, 80]. Accurate estimates of audiograms are thus essential to understanding human audition, and to all medical studies and treatments of various forms of hearing loss.

A standard auditory detection test is carried out by playing an $n$-length sequence of pure tones on a pre-defined grid in frequency-intensity space. This approach, while simple, has several salient drawbacks that lead to an unnecessarily large $n$. First, a given tone is played multiple times, even if it is highly audible or highly inaudible. Second, information is not shared between previous outcomes. For example, human audition is monotonically increasing in intensity, but in the standard test, even if a particular frequency of sound is heard at a given intensity, tones with the same frequency but higher intensity will still be tested. Finally, owing to limitations on the size of sequence $n$, a standard detection test probes only six discrete frequencies [57]. The coarseness of this grid can cause significant errors, as human hearing loss can span a range narrow enough to be entirely missed by these six frequencies [44, 115, 114]. All of these issues, combined with the impracticality and burden to human subjects of a large $n$ sequence, motivate an active learning approach.

We will take take a more modern view of audiometric testing as an active learning problem in personalized medicine [47, 78, 4], extending and adapting recent work on active learning with Gaussian processes (GPs) as discussed in Section 2.3.1 [40, 43, 26]. This approach addresses all the drawbacks of grid-sampling by performing Bayesian active learning of the audiometric function $g(\mathbf{x})$. We use this model to sequentially sample at each time step $t$ the most informative next tones conditioned on the previous $t-1$ observations $y_1, ..., y_{t-1}$. This model significantly enhances the accuracy and efficiency of learning audiograms.

These advancements stem from two major contributions that we developed after discussions with expert audiologists:

1. We extend and adapt existing work on Bayesian optimization and active learning to the setting of psychophysical detection tests. We present a model that incorporates strong prior knowledge about the auditory stimulus space, and we present experimental results demonstrating the effectiveness of a Bayesian active learning approach.
2. We develop a novel 'OR-channel' likelihood that allows the query of *multiple tones simultaneously*. We analyze this likelihood in the active learning context, clarifying the non-obvious intuition for why and when such an approach can outperform single-tone queries.

We evaluate our algorithm on both simulated and real audiometric detection tests. Our active learning approach obtains finer grained estimates of the audiogram $g(\mathbf{x})$ with substantially fewer stimuli queries (lower $n$). We note that, in the remainder of this chapter (notably our experiments), we will continue to use the example and nomenclature of audiometry, though our algorithm is precisely equivalent for other psychophysical detection tests as well.

## 3.2 Related work

A number of papers have been recently published on Bayesian active learning. Many papers have considered Bayesian active learning using mutual information in the regression setting [92, 33, 48]. However, the computation of mutual information is significantly less tractable in the classification setting. To our

knowledge, [40] is the first paper to leverage the rewriting of mutual information in (3.7), allowing for tractable computation of mutual information with the Bernoulli observation model. This paper is most similar to ours, as the Bernoulli observation model is identical to our single tone audiometric algorithm. In the conext of our application, we make two primary contributions over [40]. First, we develop a novel OR-channel likelihood, that allows for simultaneous querying of multiple tones. Second, we extend the basic mutual information derivation to this new likelihood, allowing us to compute the mutual information of an entire set of tones to be queried simultaneously. A number of other, orthogonal applications and extensions of this method have since been published [43, 26].

There has also been an enormous quantity of work on the topic of active learning outside the Bayesian active learning framework as well (e.g., [98, 51, 83, 15]). Many active learning methods even approach the problem with a similar intuition as Bayesian active learning: the strategy is to define some notion of uncertainty, and then query unlabled examples that maximize this notion of uncertainty; this notion of uncertainty is commonly based on the unlabeled example's distance from the margin. Indeed, in the Bayesian active learning framework, mutual information is also maximized when an unlabeled example is either close to the "decision boundary" $p(y \mid \mathcal{D}) = 0.5$ or has high posterior variance. While Bayesian active learning has been shown to work quite competitively with non-Bayesian appraoches, the main advantage of the Bayesian framework we focus on here is the ability to encode expert knowledge about human hearing via the mean and covariance functions. Furthermore, the Bayesian framework gives us a natural way to handle querying multiple tones simultaneously via a novel OR-channel likelhood, a concept that is not easily adapted to standard empirical risk minimization.

Alternative approaches exist for active learning as well, in particular based on expected loss reduction (e.g., [76, 34]). In these approaches, the idea is to select unlabeled examples that, in expectation over the classifier's prediction of the label, maximally reduce the loss if added to the dataset. This can in a sense be viewed as most directly analogous to the mutual information calculation used in Gaussian processes, which focus on directly reducing the entropy of the model in expectation over the label. However, using the rewriting of mutual information in (3.7), Gaussian processes enjoy the distinct advantange of being able to compute mutual information in closed form *without retraining the model*. In the audiometric setting this distinction is crucial, as we have only a small number of seconds each iteration to select a new tone, and retraining the model *for each candidate tone* is entirely infeasible.

Alternative techniques for estimating audiograms have existed for many years. Sweep-based audiometry, such as Bekesy audiometry and Audioscan, are able to produce a more continuous estimate of the audiogram that can often detect notches, but with the disadvantage of a particularly time- and attention-demanding task [44, 62]. Several Bayesian audiogram estimation techniques, such as parameter estimation by sequential testing (PEST) and maximum likelihood methods also exist, although most do not simultaneously estimate multiple frequencies [32, 54, 67, 69, 96]. More recent advances in audiometric testing have focused on improving the accessibility of hearing screening by distribution over telephone, Internet, or mobile devices [87, 93, 100, 104, 107].

## 3.3 Method

In this section, we discuss our model and approach to psychophysical detection testing using Gaussian processes. As a running example, we will use audiometry. In an audiometric detection test, a patient is presented with tones of varying frequency and intensity. The patient is asked to respond (*e.g.,* by pressing a button) if he/she hears the sound. In the absence of a timely reaction the tone is assumed to be inaudible to the patient. The delay between tones is sufficiently randomized to prevent patients from responding to predictable patterns [30].

At time step $t$, we choose a tone $\mathbf{x}_t = (\omega, i)$ with frequency $\omega$ and intensity $i$ to present to the subject. In return, we receive a response $y_t \in \{0, 1\}$, where $y_t = 1$ indicates that the patient heard the sound and $y_t = 0$ indicates that he/she did not. There is inherent observation noise in patient responses. When patients become uncertain when presented with sounds very close to their threshold (*i.e.,* the sounds become faint and hard to hear). Patients do not have perfect detection boundaries, and only hear tones near their hearing threshold with some probability. This uncertainty is observed in reality for a number of reasons. First, patient attention may waver, or they may be unable to distinguish between tones near their hearing threshold and slight background noise. Alternatively, this uncertainty may derive from physical sources. For example, if a tone is faint enough, a patient may be able to hear that tone between–but not during–heart beats. Our goal is therefore to predict the probability that a patient is able to hear a given sound.

### 3.3.1 Prior

In the case of audiometric testing, we have valuable prior knowledge about a patient's audiometric function that we can encode in our GP model. In particular, the probability that a patient hears a sound $(\omega, i)$ is *monotonically increasing* in the intensity $i$. In other words, if a tone is audible to a patient, then an even louder tone is more likely to be audible. Furthermore, audition is a *smooth function* with respect to the frequency $\omega$. Human nerves that detect similar frequencies are co-located in the cochlea and, as a result, a partial loss of hearing in one frequency is likely to cause a loss of hearing in nearby frequencies. A GP prior can encode both properties naturally through its covariance function. A combination of a linear kernel in intensity and a squared exponential kernel in frequency ensures the monotonicity and smoothness properties:

$$k\left((\omega, i), (\omega', i')\right) = ii' + \exp\left\{-\frac{1}{\ell}\|\omega - \omega'\|_2^2\right\}. \tag{3.1}$$

Here, $\ell$ regulates the smoothness (characteristic length-scale) w.r.t. frequency. Note that a GP prior is technically incapable of supporting only monotonically increasing functions. However, we only need that the posterior probability of detection be monotonic, which is generally true after a few tones are sampled (for example, see figure 3.4).

For the mean function $\mu_0$, we note that intensity is typically measured in dB HL, which is an empirical unit of measurement normalized based on population data so that at each frequency the typical human hearing threshold is around 0 dB HL. As a result we choose a constant mean function.

### 3.3.2 Observation Model

This mean function, $\mu_0(\cdot)$, and covariance function, $k(\cdot,\cdot)$, define a prior over real-valued latent functions $f \sim \mathcal{GP}(\mu_0(\cdot), k(\cdot,\cdot))$. Our goal is to predict the *probability* (*i.e.* within $[0,1]$) that a patient hears a tone with a specified frequency and intensity. We can never observe these probabilities directly. For any tone, we can instead only observe the outcome of a Bernoulli trial with the true probability. This setting is akin to Gaussian Process classification [52] and similarly we use a Bernoulli likelihood, where $\Pr[y=1|f] = \Phi(f)$ and $\Phi(\cdot)$ denotes the standard normal cumulative density function (CDF).

The linear component of the kernel in (3.1) results in a function that, after being warped by $\Phi(\cdot)$, is sigmoidal in the intensity dimension: after the slope is fixed (by conditioning on the first few points), the posterior belief about $\Phi(f)$ will tend to $0$ as the intensity decreases and $1$ as the intensity increases. This reflects our prior knowledge that tones of extremely low intensity are unlikely to be heard, whereas tones of high intensity are more likely to be audible.

**Predictions.** Once we have collected data, we can use the predictive distribution $p(y^*|X, \mathbf{y}\, \mathbf{x}^*)$ to summarize our belief about whether the patient will hear a test tone $\mathbf{x}^*$. As our likelihood is non-Gaussian, the posterior $p(f^*|X, \mathbf{y}, \mathbf{x}^*)$ has no closed form solution. However, an approximate Gaussian posterior over $f^*$ can be obtained with the standard Laplace approximation to the likelihood [52, 74].

### 3.3.3 Multiple Tones

An interesting property of audiometry (that may also be common to other psychophysical domains, *e.g.* visual or touch sensory tests), is that multiple tone stimuli can be presented to a patient simultaneously by overlaying tones. In this setting however, we can still only query whether the patient heard the overlaid tones. A negative response to a multi-tone sample indicates that the patient did not hear any of the overlaid tones; a positive response indicates that the patient heard *at least one* of them.

**OR-Channel.** Presenting a patient with $k$ tones leads to a novel extension to the standard Bernoulli likelihood used in classification. We present the patient with $k$ tones $\mathbf{x}_1, ..., \mathbf{x}_k$. The patient hearing the individual tone $\mathbf{x}_i$ is still the outcome of a Bernoulli trial with ¶$y_i|f_i = \Phi(f_i)$, as the individual trials are independent *conditioned* on $f$. However, we cannot directly observe any individual $y_i$. Rather, we record them through an *OR-channel*, that is we observe $\bar{y}$, which is $1$ if the patient hears *at least one* of the $k$ tones presented, and is $0$ otherwise. This leads to the *OR-channel likelihood*:

$$\begin{aligned} \Pr\left[\bar{y} = 1 | \mathbf{f}_{1..k}\right] &= 1 - \prod_j (1 - \Phi(f_j)) \\ &= 1 - \prod_j \Phi(-f_j) \end{aligned} \tag{3.2}$$

Note when $k = 1$, eq. (3.2) reduces to the standard Bernoulli likelihood for single tones, $\Pr\left[\bar{y} = 1 | f_1\right] = \Phi(f_1)$.

### 3.3.4 Query Selection

In iteration $t$ we present the subject with a query set of overlaid tones $\mathbf{q}_t = [\{\mathbf{x}_1, ..., \mathbf{x}_k\}]$ and query the response $\bar{y}_t$. To select $\mathbf{q}_t$ we pick the point set that maximizes the expected decrease in posterior entropy, analogous to eq. (2.9).

**Single tone mutual information.** We first consider the setting of picking a single tone, *i.e.* where $\mathbf{q}_t = [\{\mathbf{x}_t\}]$. [40] derive an analytical approximation to the mutual information, eq. (2.10), when using a Bernoulli likelihood. These results directly apply when picking a single tone $\mathbf{x}_t$. When $f_t$ is known, the entropy of the Bernoulli variable $y_t$ is given by $\mathrm{h}(\Phi(f_t))$, where

$$\mathrm{h}(p) = -p \log p - (1-p) \log(1-p),$$

is the Bernoulli entropy function. We can rephrase the entropy in eq. (2.9) as

$$I(\mathbf{f}, y_t|\mathbf{q}_t) = H\left[y_t|X, \mathbf{y}\right] - \mathbb{E}\left[H\left[y_t|\mathbf{f}\right]\right]_{p(\mathbf{f}|X,\mathbf{y})}, \tag{3.3}$$

and rewrite both terms on the right hand side through h. If $\mathbf{f}$ is unknown and $y_t$ is conditioned on $X, \mathbf{y}$, the entropy can be expressed in terms of the expectation over the posterior for $f_t$:

$$H\left[y_t|X, \mathbf{y}\right] = \mathrm{h}\left(\mathbb{E}\left[\Phi(f_t)\right]\right). \tag{3.4}$$

If $f_t$ is known we have $\Pr\left[y|\mathbf{f}\right] = \Phi(f_t)$, yielding

$$H\left[y_t|\mathbf{f}\right] = \mathrm{h}\left(\Phi(f_t)\right). \tag{3.5}$$

Substituting eqs. (3.4), (3.5) into (3.3) leads us to the following expression for the mutual information between $\mathbf{f}$ and $y_t$ in the single tone scenario:

$$I_1(\mathbf{f}, y_t|\mathbf{q}_t) = \mathrm{h}\left(\mathbb{E}\left[\Phi(f_t)\right]\right) - \mathbb{E}\left[\mathrm{h}\left(\Phi(f_t)\right)\right]. \tag{3.6}$$

The computation of $I_1$ involves an intractable integral, which can be approximated through numerical integration. This approach is very fast in practice as the integral is only one dimensional and can be computed efficiently using quadrature.

**Multiple tone mutual information** The above results can be extended to compute the mutual information when sampling multiple tones $\mathbf{q}_t = [\{\mathbf{x}_1, ..., \mathbf{x}_k\}]$. In particular, the probability of observing $\bar{y}_t = 1$ changes from $\Phi(f_t)$ to the OR-channel probability, (3.2). Thus, when $f_1, ..., f_k$ are known, the entropy of the Bernoulli variable $\bar{y}_t$ is $\mathrm{h}\left(1 - \prod_{i=1}^{k} \Phi(-f_i)\right)$.

To simplify notation, let us define $\bar{p}_1 = \Pr[\bar{y} = 1|\mathbf{f}_{1..k}]$ as defined in (3.2). Substituting $\bar{p}_1$ for $\Phi(f_t)$ in (3.6) gives the mutual information of paired tone sample $\mathbf{q}_t$ after observing the outcome $\bar{y}_t$:

$$\begin{aligned} I_k(\mathbf{f}, \bar{y}_t|\mathbf{q}_t) &= \mathrm{h}(\mathbb{E}[\bar{p}_1]) - \mathbb{E}[\mathrm{h}(\bar{p}_1)] \\ &= \mathrm{h}\Big(\mathbb{E}\left[\textstyle\prod_j \Phi(-f_j)\right]\Big) - \mathbb{E}\left[\mathrm{h}\Big(\textstyle\prod_j \Phi(-f_j)\Big)\right] \end{aligned} \tag{3.7}$$

where the second equality holds by the linearity of expectation and because $\mathrm{h}(p)$ is a concave function that is symmetric about $p = 0.5$ (*i.e.* $\mathrm{h}(p) = 1 - \mathrm{h}(p)$). The last term leads again to an intractable integral. However, similar to the one tone scenario, $I_k$ can also be evaluated efficiently using numerical integration, as $k$ is relatively small.

**Computational Considerations** Finding a set of $k \leq K$ tones $\mathbf{q}_t^{(k)}$ to maximize $I_k(\mathbf{f}, \bar{y}_t|\mathbf{q}_t)$ from a candidate set $\mathcal{X}$ of size $S$ requires $O\left(\binom{S}{k}\right)$ considerations. In order to ensure that patients do not have to wait for a lengthy duration between sounds are played, we construct a set of multiple tones to play greedily. We select

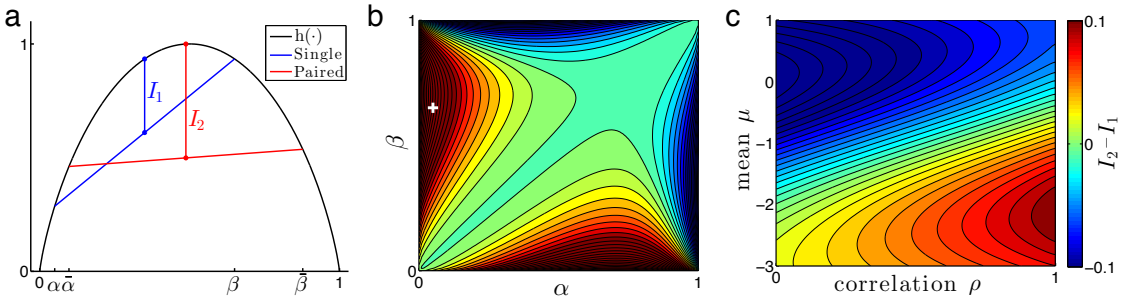


Figure 3.1: Difference in mutual information $I_2 - I_1$ between a paired query and a single query: **(a)** discrete distribution with two atoms $(\alpha, \beta) = 0.05, 0.65$, and corresponding $\bar{\alpha} = 1 - (1 - \alpha)^2 \approx 0.1$, $\bar{\beta} \approx 0.88$). Here $I_2 - I_1 \approx 0.18$; **(b)** $I_2 - I_1$ as a function of $\alpha, \beta$ (white cross denotes the specific example of panel *a*); **(c)** the normally distributed latent input case. $I_2 - I_1$ is shown as a function of the mean $\mu$ and correlation $\rho$. Colorbar at right is for both panel *b* and *c*.

the best single tone by exhaustively searching $\mathcal{X}$. Then, to select the best set of size $k$, we exhaustively add each $\hat{\mathbf{x}} \in \mathcal{X}$ to the best set of size $k-1$, $\mathbf{q}_t^{(k-1)}$ and compute the expected decrease in posterior entropy of $\mathbf{q}_t^{(k-1)} \cup \hat{\mathbf{x}}$. This greedy selection procedure reduces the computational complexity of considering tone sets of up to size $k$ to $O(Sk)$, and in practice requires only a few seconds of computation time.

### 3.3.5 Or-channel Analysis

We first investigate the OR-channel likelihood of eq. (3.2), as it is unclear if this elaboration can provide any benefit over a standard Bernoulli likelihood. Intuitively, the result of $\bar{y} = 0$ from an OR-channel is quite informative: all inputs into that channel must have been $0$ (in the auditory example, no sounds were heard). On the other hand, the result of $\bar{y} = 1$ is much less informative than in the Bernoulli channel, as it means only that one or more of the inputs were $1$ (some sound or sounds were heard), but there is no information about which. Here

we analyze simple models that support the use of the OR-channel likelihood. We compare a *single* input, corresponding to the standard Bernoulli likelihood, to a *paired* input, corresponding to an OR-channel likelihood with two inputs. That is, with inputs $\{f_1, f_2\}$ and output $y \in \{0, 1\}$ as above, our quantities of interest are $I_1 := I(y, f_1)$ and $I_2 := I(y, \{f_1, f_2\})$, and we seek to understand if more information about the inputs can exist in the paired-input query, than in the single-input query.

**OR-channel Inputs With Discrete Support**

The simplest case involves perfectly correlated inputs $f_1 = f_2$, and further, a discrete distribution on $f_1$ with two atoms of equal mass. The implied probability $\phi(f_1)$ will then have the same discrete distribution, which we write as $p\left(\phi(f_1)\right) = \frac{1}{2}\delta(\phi(f_1) = \alpha) + \frac{1}{2}\delta(\phi(f_1) = \beta)$, for some atoms $\alpha$ and $\beta$. Then, the mutual information of the single query is:

$$\begin{aligned} I_1 &= H(y) - H(y|f_1) \\ &= \mathrm{h}\left(\mathbb{E}_f\left[\phi\left(f_1\right)\right]\right) - \mathbb{E}_f\left[\mathrm{h}\left(\phi\left(f_1\right)\right)\right] \\ &= \mathrm{h}\left(\frac{1}{2}(\alpha+\beta)\right) - \frac{1}{2}\left(\mathrm{h}(\alpha) + \mathrm{h}(\beta)\right), \end{aligned} \tag{3.8}$$

where $\mathbb{E}_f$ is the expectation under the distribution on $f$. The OR-channel likelihood for two terms is similarly $p(y = 1|\{f_1, f_2\}) = 1 - (1 - \phi(f_1))\,(1 - \phi(f_2)) = 1 - (1 - \phi(f_1))^2$. The mutual information of a paired-input query becomes

$$I_2 = \mathrm{h}\left(\frac{1}{2}\left(\bar{\alpha} + \bar{\beta}\right)\right) - \frac{1}{2}\left(\mathrm{h}\left(\bar{\alpha}\right) + \mathrm{h}\left(\bar{\beta}\right)\right), \tag{3.9}$$

where $\bar{\alpha} = 1 - (1-\alpha)^2$ and $\bar{\beta} = 1 - (1-\beta)^2$. $I_2$ and $I_1$ offer a convenient geometric interpretation by viewing mutual information as the Jensen's inequality gap of h (eqs. (3.8) and (3.9)). With this simple discrete distribution, $\alpha$ and $\beta$ can be chosen

such that $I_2 - I_1$ will be positive or negative. We show the critical case $I_2 > I_1$ in Figure 3.1a, where the blue line segment connects $(\alpha, \mathrm{h}(\alpha))$ to $(\beta, \mathrm{h}(\beta))$ with $(\alpha, \beta) = (0.05, 0.65)$, and the red line segment is then implied by those choices of $\alpha, \beta$ (that is, $(\bar{\alpha}, \bar{\beta}) \approx (0.10, 0.88)$ in the figure). Here the difference is $I_2 - I_1 = 0.18$ bits. The contours of $I_2 - I_1$ as a function of $(\alpha, \beta)$ is shown in Figure 3.1b.

**OR-channel Inputs With Normal Densities**

We next analyze the OR-channel likelihood with two latent factors $f_1 = f(x_1)$ and $f_2 = f(x_2)$, which are jointly Gaussian according to the GP model of Section 3.3: $[f_1, f_2] \sim \mathcal{N}(m, S)$. We calculate $I_2 - I_1$ numerically using eq. (3.6) (note that, compared to the previous example, only the expectation over $f$ has changed). We simplify the parameter space with $m = \begin{bmatrix} \mu \\ \mu \end{bmatrix}$ and $S = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$ (but note that the function $I_2 - I_1$ is not invariant to either of these simplifications). We plot the contours of $I_2 - I_1$ as a function of correlation $\rho$ and mean $\mu$ in Figure 3.1c, which indeed has substantial regions of both positive and negative mass.

In summary, though intuitively non-obvious, the above analyses clarify that the OR-channel likelihood can, but need not, increase mutual information between the input distribution and the binary outcome $y$. This finding offers a critical takeaway: the OR-channel can be used effectively, but only in the setting where a judicious choice of input distribution can be made. Indeed, this is exactly what our framework will achieve: it will choose pairs of input points (paired sounds) to learn more about the underlying audiogram than a single point alone. Thus, the OR-channel likelihood offers benefit beyond this scheme, which we already expect to outperform a naive approach to learning these la-

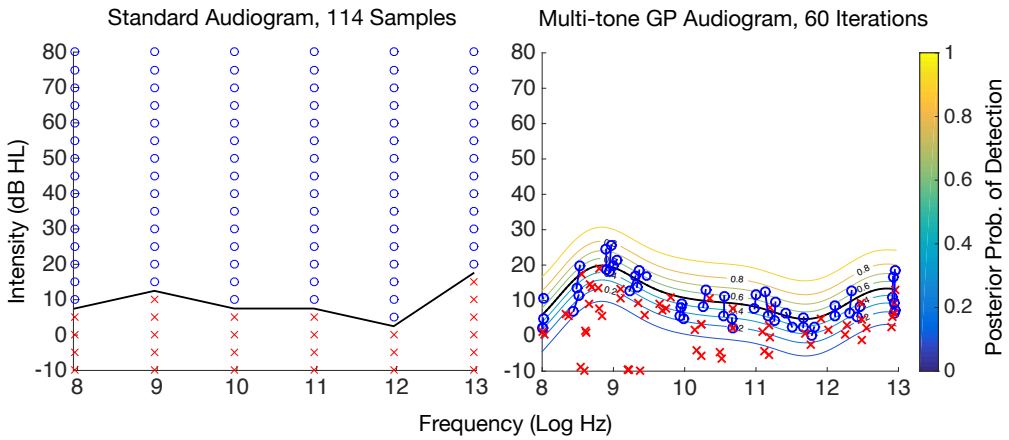


Figure 3.2: Standard grid search audiogram with tones played at every octave from 250 to 8000 Hz, and every 5 dB HL from -10 dB to 80 dB, compared to a multi-tone GP audiogram with 60 iterations (and therefore 119 "samples").

tent functions. In this work we only consider paired inputs; a future question for study is how the information gain distribution changes with increasing numbers of inputs.

## 3.4 Results

In this section, we empirically evaluate our proposed algorithms for psychophysical detection. We focus on our application to audiometry, and seek to evaluate the merits of using Gaussian processes for audiometry in general, as well as to compare single-tone and multi-tone audiometry, focusing on the machine learning aspects of our algorithms. We have also evaluated our method further in a clinical trial [91], which we discuss further in the discussion section.

To begin, we compare the audiograms found by a standard grid audiometric test and by our multi-tone GP model. In both cases we run the same human subject in the same audiometric setting. The only differences are the tones presented

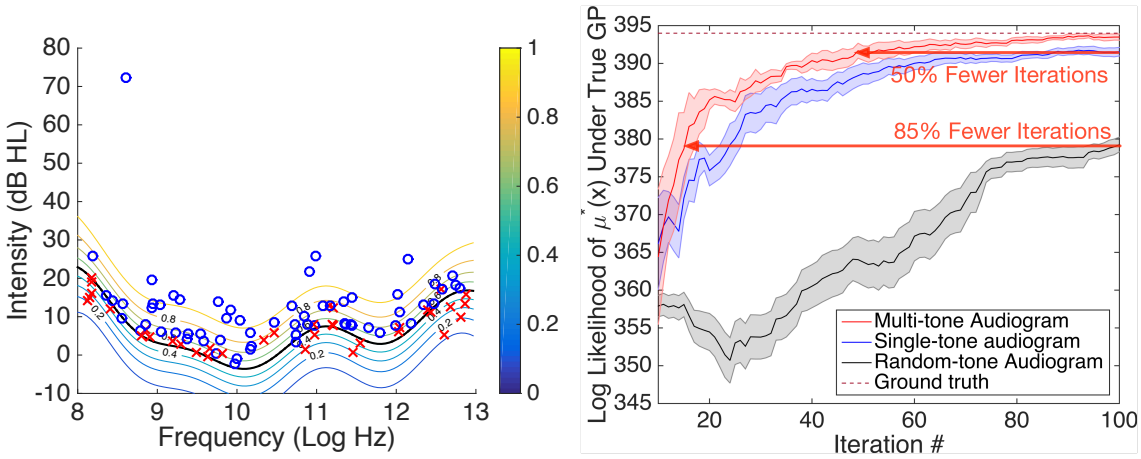


Figure 3.3: Comparison of multi-tone and single-tone GP audiometrics. **Left:** A GP trained on 100 single tones. Blue circles denote tones detected by the subject, and red crosses denote tones that were not detected. The posterior probabilities are shown as color contours. **Right:** Log likelihood of random presentation of tones (no active learning, shown in gray), active learning presentation of single tones (shown in blue), and active learning with paired tones (shown in red), under the ground truth audiometric function from the left figure. Log likelihood is plotted as a function of iterations in each audiometric testing strategy. Shaded areas denote standard error.

and the method used to infer the audiometric function. All audiometric tests were run in accordance with an approved IRB. In the standard setting, tones from this grid are presented in a pre-determined order, typically ascending in frequency and decreasing in intensity. In the GP model, pairs of tones were actively selected given all previous pairs of tones and the responses to those tones. A random delay of up to 3 seconds was inserted between tone presentations to prevent subjects from memorizing a pattern in the test. Figure 3.2 shows the resulting data and inferred audiograms plotted in frequency-intensity space (left panel: standard audiometric test; right panel: GP method). For both the standard and GP experiments, tones that were detected by the patient are plotted as blue circles, and tones that were not detected are plotted as red crosses. For the paired-tone GP test (right panel), paired samples that were detected are plotted as blue

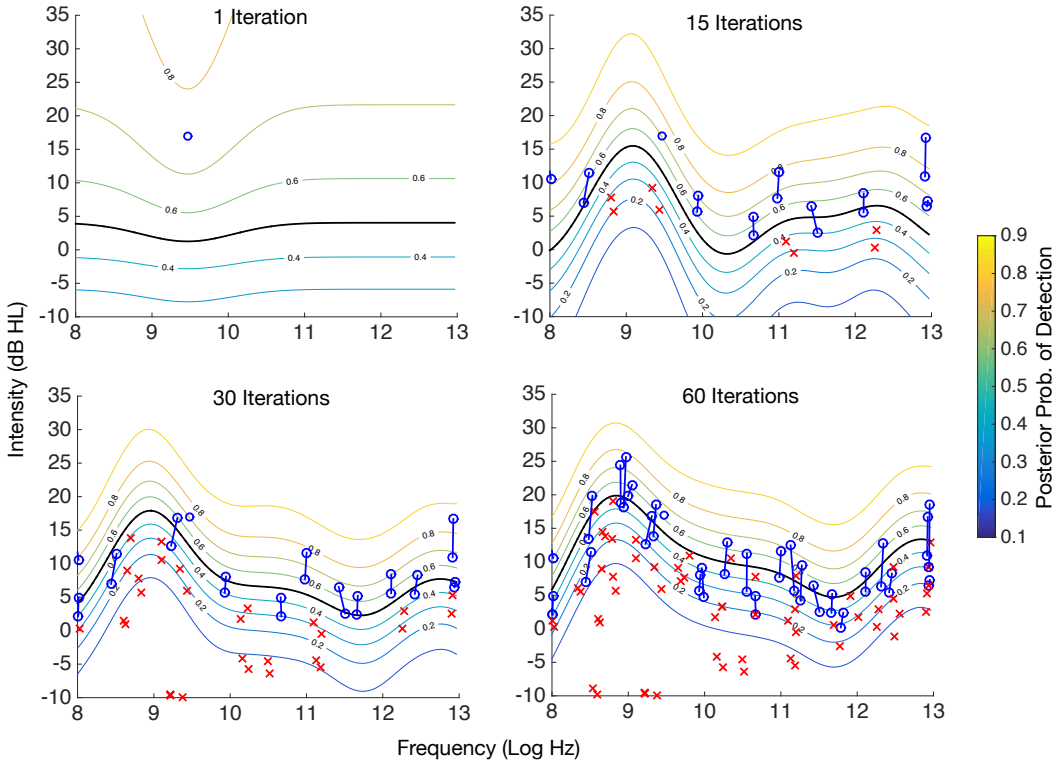


Figure 3.4: The posterior probability of detection within the frequency / intensity space during a GP audiometric test on a human subject. Panels show the learned GP after 1, 15, 30, and 60 iterations. Queries consist of a single or a paired tone (as selected by the model). Blue circles indicate a positive outcome (sound was heard), red crosses indicate a negative outcome. Paired tones with positive outcome (at least one of the two tones was heard) are connected by a blue line. Almost all queries are close to the final audible threshold ($0.5$ posterior detection probability), which is well approximated even after only $15$ iterations.

circles connected by a blue line (recall that, due to the OR-channel likelihood, we do not know which tone was heard). Paired tones that were not detected are again plotted as individual red crosses, as these data are functionally equivalent to two single-tone samples that were not detected (again due to the OR-channel observation model).

In the standard audiometric test, the inferred audiogram is simply an "audible threshold" that is the piecewise linear function connecting the detection

threshold at each frequency. This threshold is depicted as a black line in the left panel of Figure 3.2. In the GP case, we infer a full posterior distribution on the detection threshold. We plot contours of the posterior detection probability in the right panel of Figure 3.2, with a solid black line at $50\%$ posterior detection probability.

This confirmatory comparison offers several key points of interpretation. First, the tests agree with each other: the $50\%$ posterior detection probability in the GP case is within 5dB of the standard audiogram, giving confidence to the general sensibility of this model. Second, perhaps most importantly to the active learning goal, the GP active learning model presents approximately half as many iterations (60 actively learned paired tones compared to 114 single tones preselected from a grid). Thus the GP model is able to explore substantially more of the frequency space than the standard grid test, and it does so in many fewer overall iterations, reducing the burden of these tests. Third, note that the GP model does not explore uninformative regions of tone space: above a certain intensity (at which the model is confident that tones are certainly heard), there are no tones queried. This observation differs sharply from the standard test, which squanders numerous samples at intensities well above this subject's audible threshold, where little to no information is available. Fourth, by design our GP model offers a full posterior distribution over tone space, and thus produces a richer and more descriptive audiogram than the piecewise linear audible threshold function in the standard test. Finally, it is worth noting that, though the paired tones in the right panel of Figure 3.2 appear to be sampled at very similar frequencies in log-space, the differences were often nontrivial, up to four or five half steps in an octave.

Next, Figure 3.4 investigates the convergence of our GP model after $1, 15, 30, 60$ iterations of our paired-tone GP audiometric algorithm. The posterior after a single iteration (upper left panel) reflects primarily the prior mean and the covariance of the model, which incorporates our knowledge about the general shape of human audiograms. As the active learning procedure continues (other panels), the GP posterior quickly converges to the audiogram of this particular subject. After only 30 iterations, the GP model has already captured the audiogram shape, and subsequent changes are very minor.

To investigate the performance of our GP active learning method in greater detail, we construct a synthetic data set with known ground truth (a known audiometric function). We begin by training a GP on $100$ single tones and the detection of those tones reported by a second human subject. The tones sampled and the inferred audiogram are presented in Figure 3.4. We use this posterior GP as the true audiogram of a simulated subject.

This ground truth audiometric function allows for the critical assessment of performance shown in Figure 3.4. We compare three strategies of data presentation: random presentation of tones (no active learning, shown in gray), active learning presentation of single tones (shown in blue), and active learning with paired tones (shown in red). For each strategy, at each iteration (tone presentation), we infer the GP posterior mean, which is the MAP estimate of the audiometric function, given each stream of data. We evaluate the log likelihood of each strategy's GP posterior mean under the ground truth GP from Figure 3.4. This step offers a quantitative assessment of how closely each strategy has approximated the true audiometric function. The maroon dashed line depicts the log likelihood of the ground truth GP itself, which is thus the maximum achiev-

able performance of any strategy. All three strategies (random, single tone active learning, paired tone active learning) should, with enough iterations, converge to ground truth. Thus, the essential question of this work, and indeed of any active learning method, is how much more quickly a particular strategy approaches the ground truth than competing strategies.

We ran the single and paired tone active learning methods ten times each, and standard errors are plotted as shaded regions. Because of the very high standard error of the random tone audiogram, these results were averaged over 100 runs.

Figure 3.4 has a few key findings. Both the single and paired tone active learning strategies significantly outperform random sampling. Thus our strong prior rapidly learns that large portions of the tone space are either very likely or very unlikely to be heard, and is able to quickly learn to sample in regions of high information. After 80-90 iterations the paired tone algorithm matches the ground truth model very closely. This result is in significant contrast to randomly choosing tones, which not only has very large standard error, but also rarely converges to a good model. Finally, we observe that the paired tone active learning strategy significantly outperforms the single tone strategy. In fact, the paired tone strategy requires only half as many iterations to achieve the same level of likelihood. Compared to random sampling, paired tone active learning reduces the number of iterations by $85\%$.

## 3.5 Discussion

In this chapter, we explored the problem of adapting Bayesian active learning to psychophysical testing, and improving upon standard techniques used in audiometric testing. The success of our method is due primarily to the incorporation of problem specific information in the Gaussian process model. By using a linear kernel to model the simple fact that human hearing is monotonic with volume, our audiometric test avoids exploring large portions of the audiometric test space corresponding to tones louder than ones already heard by the patient.

Additionally, we developed a novel OR-channel likelihood that allows us to present multiple tones to a subject simultaneously, leading to an audiometric testing strategy that not only yields good audiogram estimation using significantly fewer samples, but also leads to much better coverage of the frequency dimension. We demonstrate a non-obvious result, that multiple tones played through an OR-channel can, but do not have to, yield more information than a single tone.

**Clinical results.** As a consequence of the promising results for the methods discussed in this paper, our co-authors have conducted a clinical trial of these methods in [91]. In that paper, these methods were evaluated in 21 participants between the ages of 18 and 90 years old, with varying degrees of hearing capability. It was found that the methods in this paper do indeed provide comparable or better audiometric testing results to the standard audiometric test, but with significantly fewer tones presented to the patient. However, matching the standard audiometric test in some sense is only the most basic advantage of our method. In particular, the fully probabilistic treatment of human hearing is novel to this

paper, and allows practioners to more precisely reason about patient hearing. Pure tone audiometry is inherently noisy and prone to subject error, particularly when presenting tones around the patient's hearing threshold.

**Adoption of our method.** This work was done in collaboration in part with Dennis Barbour, a medical expert specializing in part in audiometry. He has made substantial improvements to the work since its publication, particularly in terms of its ease of use by medical experts rather than machine learning experts, and these methods are deployed in a number of real-world settings where they are used to get fast and accurate audiograms.

**Future directions.** From a purely application-oriented stand point, one of the most obvious future directions is the application of these techniques to other psychophysical detection tests than the audiometric test. In particular, there is strong reason to believe that existing methods used for certain types of visual and touch-based tests could be dramatically improved both by incorporating strong prior knowledge about patients. Furthermore, the ability to present patients with multiple stimuli simultaneously is inherent to many types of psychophysical tests. For example, when testing a patient for loss of sense of touch, the OR-channel model applies directly: a patient can be presented with multiple touch stimuli, and can easily respond whether *at least one* stimulus was felt.

CHAPTER 4

# BAYESIAN ACTIVE MODEL SELECTION

## 4.1 Introduction

In the previous chapter, we detailed a case study of a real world application where incorporating strong prior knowledge using Gaussian processes has a dramatic impact on performance. By encoding *a priori* in to the model some simple facts about human hearing–that people are more likely to hear loud tones than soft ones, and that hearing loss in a particular frequency strongly implies hearing loss in nearby frequencies–we were able to dramatically improve the standard audiometric detection test.

The work presented in Chapter 3 additionally poses the following question: can we *automate* the process of encoding prior knowledge? This process can be quite difficult, as it requires expert knowledge in both the problem domain and in Gaussian processes in order to construct a prior that best models the problem. When sufficient data is available, techniques based on Bayesian model selection (e.g., [19]) offer a natural solution to this problem: the best prior is the one with the highest model posterior after conditioning on the data–or even better, an average over all available models weighted by the model posterior.

However, in many scenarios like the audiometric setting, the problem by definition afford limited or no data at all: the goal is to *actively acquire* data to learn the function of interest (for example, the patient's audiometric function) as quickly as possible. This leads naturally to the question: can we additionally actively acquire data to learn *which prior or model best explains* the function of

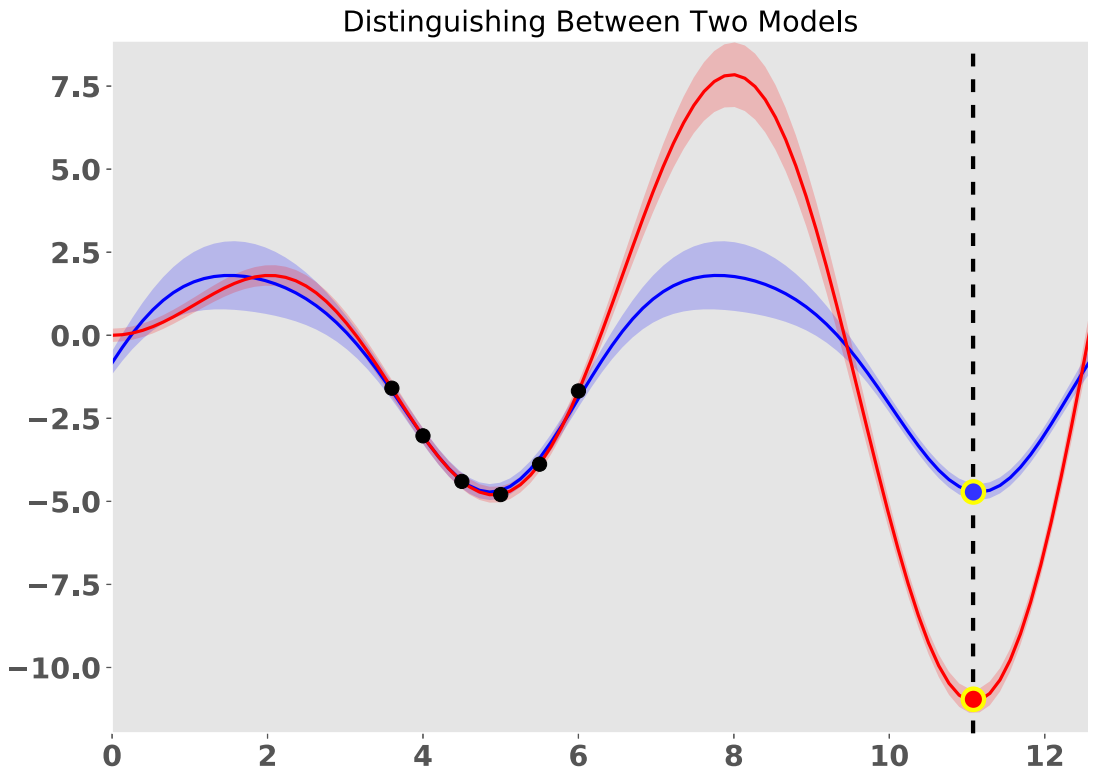


Figure 4.1: Distinguishing two models with active model selection. Given only the input data (black dots), deciding whether the red model or the blue model best describes the true function is a difficult task. However, with a single additional training input at the dashed black line, the models are immediately distinguished: the potential blue point has very low probability under the red model, and the potential red point has very low probability under the blue model. Choosing the next point that maximizes the disagreement among all candidate models is achieved by maximizing the *mutual information*.

interest?

A particular, clarifying example that motivates this work is *noise-induced hearing loss* (NIHL), a prevalent disorder affecting 26 million working-age adults in the United States alone [84] and affecting over half of workers in particular occupations such as mining and construction. Most tragically, NIHL is entirely preventable with simple, low-cost solutions (e.g., earplugs). The critical requirement for prevention is effective early diagnosis.

To be tested for NIHL, patients must complete a time-consuming audiometric exam that presents a series of tones at various frequencies and intensities; at each tone the patient indicates whether he/she hears the tone [11, 41, 16]. From the responses, the clinician infers the patient's audible threshold on a set of discrete frequencies (the *audiogram*); this process requires the delivery of up to hundreds of tones. Audiologists scan the audiogram for a hearing deficit with a characteristic *notch* shape—a narrow band that can be anywhere in the frequency domain that is indicative of NIHL. Unfortunately, at early stages of the disorder, notches can be small enough that they are undetectable in a standard audiogram, leaving many cases undiagnosed until the condition has become severe. Increasing audiogram resolution would require higher sample counts (more presented tones) and thus only lengthen an already burdensome procedure. We present here a better approach.

Note that the NIHL diagnostic challenge is not one of feature selection (choosing the next test to run and classifying the result), but rather of model selection: is this patient's hearing better described by a normal hearing model, or a notched NIHL model? Here we propose a novel *active model selection* algorithm to make the NIHL diagnosis in as few tones as possible, which directly reflects the time and personnel resources required to make accurate diagnoses in large populations. We note that this is a model-selection problem in the truest sense: a diagnosis corresponds to selecting between two or more sets of indexed probability distributions (models), rather than the more-common misnomer of choosing an index from within a model (i.e., hyperparameter optimization). In the NIHL case this distinction is critical. We are choosing between two models, the set of possible NIHL hearing functions and the set of normal hearing functions. This approach suggests a very different and more direct algorithm than first learning

the most likely NIHL function and then accepting or rejecting it as different from normal, the standard approach.

We make the following contributions: first, we design a completely general active-model-selection method based on maximizing the mutual information between the response to a tone and the posterior on the model class. Critically, we develop an analytical approximation of this criterion for Gaussian process (GP) models with arbitrary observation likelihoods, enabling active structure learning for GPs. Second, we extend the work of [25] (which uses active learning to speed up audiogram inference) to the broader question of identifying *which model*—normal or NIHL—best fits a given patient. Finally, we develop a novel GP prior mean that parameterizes notched hearing loss for NIHL patients. To our knowledge, this is the first publication with an active model-selection approach that does not require updating each model for every candidate point, allowing audiometric diagnosis of NIHL to be performed in real time. Finally, using patient data from a clinical trial, we show empirically that our method typically automatically detects simulated noise-induced hearing loss with fewer than 15 query tones. This is vastly fewer than the number required to infer a conventional audiogram or even an actively learned audiogram [25], highlighting the importance of both the active-learning approach and our focus on model selection.

### 4.1.1 Related work

Although active learning and model selection have been widely investigated, active model selection has received comparatively less attention. [1] proposed

an active learning model selection method that requires leave-two-out cross validation when evaluating each candidate $\mathbf{m}\mathrm{x}^*$, requiring $\mathcal{O}\big(B^2M|\mathbf{m}\mathrm{X}^*|\big)$ model updates per iteration, where $B$ is the total budget. [50] also considered an information-theoretic approach to active model selection, suggesting maximizing the expected cross entropy between the current model posterior $p(\mathcal{M} \mid \mathcal{D})$ and the updated distribution $p(\mathcal{M} \mid \mathcal{D}')$. This approach also requires extensive model retraining, with $\mathcal{O}\big(M|\mathbf{m}\mathrm{X}^*|\big)$ model updates per iteration, to estimate this expectation for each candidate. These approaches become prohibitively expensive for real-time applications with large number of candidates. In our audiometric experiments, for example, we consider 10 000 candidate points, expending 1–2 seconds per iteration, whereas these mentioned techniques would take several hours to selected the next point to query.

An alternative approach to dealing with uncertainty about what model (kernel) should be used is to choose an extremely flexible, highly parametric kernel and learn the structure of data via gradient descent. This is an approach taken by multiple kernel learning [29] and recently using the spectral mixture kernel [111] or deep kernel learning [112, 110]. The primary advantage of composing simple kernels as done in for example [19] is that these highly parametric kernels typically require a large number of training examples to learn useful kernels. In the active learning regime, the goal is to specifically finish learning *before* collecting enough data for these techniques to be useful. As a result, we focus only on kernels with a small number of hyperparameters.

## 4.2 Active Bayesian model selection

Suppose that we have a mechanism for actively selecting new data—choosing $\mathbf{x}^* \in \mathcal{X}$ and observing $y^* = y(\mathbf{x}^*)$—to add to our dataset $\mathcal{D} = (X, \mathbf{y})$, in order to better distinguish the candidate models $\{\mathcal{M}_i\}$. After making this observation, we will form an augmented dataset $\mathcal{D}' = \mathcal{D} \cup \{(\mathbf{x}^*, \mathbf{y}^*)\}$, from which we can recompute a new model posterior $p(\mathcal{M} \mid \mathcal{D}')$.

An approach motivated by information theory is to select the location maximizing the *mutual information* between the observation value $y^*$ and the unknown model:

$$I(y^*; \mathcal{M} \mid \mathbf{x}^*, \mathcal{D}) = H[\mathcal{M} \mid \mathcal{D}] - \mathbb{E}_{y^*}\big[H[\mathcal{M} \mid \mathcal{D}']\big] \tag{4.1}$$

$$= H[y^* \mid \mathbf{x}^*, \mathcal{D}] - \mathbb{E}_{\mathcal{M}}\big[H[y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}]\big], \tag{4.2}$$

where $H$ indicates (differential) entropy. Whereas Equation (4.1) is computationally problematic (involving costly model retraining), the equivalent expression (4.2) is typically more tractable, has been applied fruitfully in various active-learning settings [40, 26, 25, 38, 39], and requires only computing the differential entropy of the model-marginal predictive distribution:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}) = \sum_{i=1}^{M} p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i) p(\mathcal{M}_i \mid \mathcal{D}) \tag{4.3}$$

and the model-conditional predictive distributions $\{p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i)\}$ with all models trained with the currently available data. In contrast to (4.1), this does not involve any retraining cost. Although computing the entropy in (4.3) might be problematic, we note that this is a one-dimensional integral that can easily be resolved with quadrature. Our proposed approach, which we call *Bayesian active model selection* (BAMS) is then to compute, for each candidate location $\mathbf{x}^*$,

the mutual information between $y^*$ and the unknown model, and query where this is maximized:

$$\arg\max_{\mathbf{x}^*} I(y^*; \mathcal{M} \mid \mathbf{x}^*, \mathcal{D}). \tag{4.4}$$

## 4.3 Active model selection for Gaussian processes

In the previous section, we proposed a general framework for performing sequential active Bayesian model selection, without making any assumptions about the forms of the models $\{\mathcal{M}_i\}$. Here we will discuss specific details of our proposal when these models represent alternative structures for Gaussian process priors on a latent function.

We assume that our observations are generated via a latent function $f : \mathcal{X} \to \mathbb{R}$ with a known observation model $p(\mathbf{y} \mid \mathbf{f})$, where $f_i = f(\mathbf{x}_i)$. We place a Gaussian process (GP) prior distribution on $f$, $p(f) = \mathcal{GP}(f; \mu, K)$

As discussed in Chapter 2, the *structural* (not hyperparameter) choices made in the mean function $\mu$ and covariance function $K$ themselves are typically done by selecting (often blindly!) from several off-the-shelf solutions (see, for example, [18, 72]; though also see [20, 109]), and this choice has substantial bearing on the resulting functions $f$ we can model. Indeed, in many settings, choosing the nature of plausible functions is precisely the problem of model selection; for example, to decide whether the function has periodic structure, exhibits nonstationarity, etc. Our goal is to automatically and actively decide these structural choices during GP modeling through intelligent sampling.

To connect to our active learning formulation, let $\{\mathcal{M}_i\}$ be a set of Gaussian

process models for the latent function $f$. Each model comprises a mean function $\mu_i$, covariance function $K_i$, and associated hyperparameters $\theta_i$. Our approach outlined in Section 4.2 requires the computation of three quantities that are not typically encountered in GP modeling and inference: the hyperparameter posterior $p(\theta \mid \mathcal{D}, \mathcal{M})$, the model evidence $p(\mathbf{y} \mid X, \mathcal{M})$, and the predictive distribution $p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M})$, where we have marginalized over $\theta$ in the latter two quantities. The most-common approaches to GP inference are maximum likelihood–II (MLE) or maximum *a posteriori*–II (MAP) estimation, where we maximize the hyperparameter posterior [106, 72]:[1]

$$\hat{\theta} = \arg\max_{\theta} \log p(\theta \mid \mathcal{D}, \mathcal{M}) = \arg\max_{\theta} \log p(\theta \mid \mathcal{M}) + \log(\mathbf{y} \mid X, \theta, \mathcal{M}). \quad (4.5)$$

Typically, predictive distributions and other desired quantities are then reported at the MLE / MAP hyperparameters, implicitly making the assumption that $p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \delta(\hat{\theta})$. Although a computationally convenient choice, this does not account for uncertainty in the hyperparameters, which can be nontrivial with small datasets [56]. Furthermore, accounting correctly for model parameter uncertainty is crucial to model selection, where it naturally introduces a model-complexity penalty. We discuss less-drastic approximations to these quantities below.

### 4.3.1 Approximating the model evidence and hyperparameter posterior

The model evidence $p(\mathbf{y} \mid X, \mathcal{M})$ and hyperparameter posterior distribution $p(\theta \mid \mathcal{D}, \mathcal{M})$ are in general intractable for GPs, as there is no conjugate prior

[1] Using a noninformative prior $p(\theta \mid \mathcal{M}) \propto 1$ in the case of maximum likelihood.

distribution $p(\theta \mid \mathcal{M})$ available. Instead, we will use a Laplace approximation, where we make a second-order Taylor expansion of $\log p(\theta \mid \mathcal{D}, \mathcal{M})$ around its mode $\hat{\theta}$ (4.5). The result is a multivariate Gaussian approximation:

$$p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \mathcal{N}(\theta; \hat{\theta}, \Sigma); \qquad \Sigma^{-1} = -\nabla^2 \log p(\theta \mid \mathcal{D}, \mathcal{M})\big|_{\theta=\hat{\theta}}. \tag{4.6}$$

The Laplace approximation also results in an approximation to the model evidence:

$$\log p(\mathbf{y} \mid X, \mathcal{M}) \approx \log p(\mathbf{y} \mid X, \hat{\theta}, \mathcal{M}) + \log p(\hat{\theta} \mid \mathcal{M}) - \tfrac{1}{2} \log \det \Sigma^{-1} + \tfrac{d}{2} \log 2\pi, \tag{4.7}$$

where $d$ is the dimension of $\theta$ [71, 49]. The Laplace approximation to the model evidence can be interpreted as rewarding explaining the data well while penalizing model complexity. Note that the *Bayesian information criterion* (BIC), commonly used for model selection, can be seen as an approximation to the Laplace approximation [81, 65].

### 4.3.2 Approximating the predictive distribution

We next consider the predictive distribution:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}) = \int p(y^* \mid f^*) \underbrace{\int p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}) p(\theta \mid \mathcal{D}, \mathcal{M}) \, \mathrm{d}\theta}_{p(f^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M})} \, \mathrm{d}f^*. \tag{4.8}$$

The posterior $p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M})$ in (4.8) is typically a known Gaussian distribution, derived analytically for Gaussian observation likelihoods or approximately using standard approximate GP inference techniques [52, 63]. However, the integral over $\theta$ in (4.8) is intractible, even with a Gaussian approximation to the hyperparameter posterior as in (4.6).

Garnett et al., 2013 introduced a mechanism for approximately marginalizing GP hyperparameters (called the MGP), which we will adopt here due to its strong empirical performance. The MGP assumes that we have a Gaussian approximation to the hyperparameter posterior, $p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \mathcal{N}(\theta; \hat{\theta}, \Sigma)$.[2] We define the posterior predictive mean and variance functions as

$$\mu^*(\theta) = \mathbb{E}[f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}]; \qquad \nu^*(\theta) = \mathrm{Var}\left[[]\, f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}\right].$$

The MGP works by making an expansion of the predictive distribution around the posterior mean hyperparameters $\hat{\theta}$. The nature of this expansion is chosen so as to match various derivatives of the true predictive distribution; see [26] for details. The posterior distribution of $f^*$ is approximated by

$$p(f^* \mid \mathbf{m}\mathrm{x}^*, \mathcal{D}, \mathcal{M}) \approx \mathcal{N}\big(f^*; \mu^*(\hat{\theta}), \sigma^2_{\mathrm{MGP}}\big), \tag{4.9}$$

where

$$\sigma^2_{\mathrm{MGP}} = \tfrac{4}{3}\nu^*(\hat{\theta}) + \big[\nabla\mu^*(\hat{\theta})\big]^\top \Sigma \big[\nabla\mu^*(\hat{\theta})\big] + \tfrac{1}{3\nu^*(\hat{\theta})}\big[\nabla\nu^*(\hat{\theta})\big]^\top \Sigma \big[\nabla\nu^*(\hat{\theta})\big]. \tag{4.10}$$

The MGP thus inflates the predictive variance from the the posterior mean hyperparameters $\hat{\theta}$ by a term that is commensurate with the uncertainty in $\theta$, measured by the posterior covariance $\Sigma$, and the dependence of the latent predictive mean and variance on $\theta$, measured by the gradients $\nabla\mu^*$ and $\nabla\nu^*$. With the Gaussian approximation in (4.9), the integral in (4.8) now reduces to integrating the observation likelihood against a univariate Gaussian. This integral is often analytic [72] and at worse requires one-dimensional quadrature.

[2] This is arbitrary and need not be the Laplace approximation in (4.6), so this is a slight abuse of notation.

### 4.3.3 Implementation

Given the development above, we may now efficiently compute an approximation to the BAMS criterion for active GP model selection. Given currently observed data $\mathcal{D}$, for each of our candidate models $\mathcal{M}_i$, we first find the Laplace approximation to the hyperparameter posterior (4.6) and model evidence (4.7). Given the approximations to the model evidence, we may compute an approximation to the model posterior (2.13). Suppose we have a set of candidate points $X^*$ from which we may select our next point. For each of our models, we compute the MGP approximation (4.9) to the latent posteriors $\{p(\mathbf{f}^* \mid X^*, \mathcal{D}, \mathcal{M}_i)\}$, from which we use standard techniques to compute the predictive distributions $\{p(\mathbf{y}^* \mid X^*, \mathcal{D}, \mathcal{M}_i)\}$. Finally, with the ability to compute the differential entropies of these model-conditional predictive distributions, as well as the marginal predictive distribution (4.3), we may compute the mutual information of each candidate in parallel.

## 4.4 Computational details for common observation likelihoods

Here we give further details for computing Equation (4.2) with common observation likelihoods.

### 4.4.1 Regression with Gaussian noise

For regression problems with zero-mean homoskedastic Gaussian noise with variance $\sigma_n^2$, we have

$$p(y \mid f) = \mathcal{N}(y; f, \sigma_n^2).$$

We may integrate this against a Gaussian distribution on the latent value $f$ (such as that resulting from the MGP approximation (4.9)) to find the predictive distribution. Suppose $p(f \mid \mathbf{m}\mathrm{x}, \mathcal{D}) = \mathcal{N}(f; \mu, \sigma^2)$. Then

$$p(y \mid \mathbf{m}\mathrm{x}, \mathcal{D}) = \mathcal{N}(y; \mu, \sigma^2 + \sigma_n^2).$$

The model-conditional predictive distributions $\{p(y^* \mid X^*, \mathcal{D}, \mathcal{M}_i)\}$ are thus Gaussian under the MGP approximation:

$$p(y^* \mid \mathbf{m}\mathrm{x}^*, \mathcal{D}, \mathcal{M}_i) = \mathcal{N}\big(y^*; \mu_i^*, (\nu_{\mathrm{MGP}}^*)_i + \sigma_n^2\big).$$

The differential entropy of a one-dimensional Gaussian is

$$H\big[\mathcal{N}(y; \mu, \sigma^2)\big] = \tfrac{1}{2}\log(2\pi e\sigma^2);$$

with these results and an approximation to the model posterior, we may compute the second term in (4.2). The marginal predictive distribution $p(y^* \mid \mathbf{m}\mathrm{x}^*, \mathcal{D})$ is now a mixture of Gaussians weighted by the approximate model posterior. The differential entropy of a mixture of Gaussians does not have a closed form, so for the first term in (4.2), we must resort to (one-dimensional) quadrature.

Note that we may also treat the noise variance $\sigma_n^2$ as a model-dependent hyperparameter and use the MGP to approximately marginalize it as well, which changes the above only slightly. An extension to heteroskedastic noise is also trivial.

### 4.4.2 Probit regression

For binary classification problems with a probit likelihood, we have

$$\Pr\left[(\right] y = 1 \mid f) = \Phi(f),$$

where $\Phi$ is the univariate standard normal CDF. We may integrate this against a Gaussian distribution on the latent value $f$ to find the predictive distribution. Suppose $p(f \mid \mathbf{m}\mathrm{x}, \mathcal{D}) = \mathcal{N}(f; \mu, \sigma^2)$. Then

$$\Pr\left[(\right] y = 1 \mid \mathbf{m}\mathrm{x}, \mathcal{D}) = \int \Phi(f)\mathcal{N}(f; \mu, \sigma^2)\,\mathrm{d}f = \Phi\left(\frac{\mu}{\sqrt{1+\sigma^2}}\right).$$

The model-conditional predictive distributions $\left\{p(y^* \mid X^*, \mathcal{D}, \mathcal{M}_i)\right\}$ are thus Bernoulli distributions under the MGP approximation:

$$p(y^* \mid \mathbf{m}\mathrm{x}^*, \mathcal{D}, \mathcal{M}_i) = \mathcal{B}\left(\Phi\left(\frac{\mu_i^*}{\sqrt{1+(\nu_{\mathrm{MGP}}^*)_i}}\right)\right).$$

The differential entropy of a Bernoulli distribution with success probability $p$ is given by the Bernoulli entropy function $h$:

$$H\left[\mathcal{B}(p)\right] = h(p) = -p\log p - (1-p)\log(1-p);$$

with these results and an approximation to the model posterior, we may compute the second term in (4.2). The marginal predictive distribution $p(y^* \mid \mathbf{m}\mathrm{x}^*, \mathcal{D})$ is now a mixture of Bernoullis weighted by the approximate model posterior. Bernoulli distributions are closed under taking mixtures, and therefore in this case we may compute both terms in (4.2) without resorting to quadrature.

## 4.5 Audiometric threshold testing

Noise-induced hearing loss occurs when an otherwise healthy individual is habitually subjected to high-intensity sound [61]. This differs substantially from the

standard audiometric setting discussed in Chapter 3 because this type of hearing loss can present as a sharp, notch-shaped hearing loss in a narrow (sometimes less than one octave) frequency range. From a technical perspective of the model presented in Chapter 3, this violates the smoothness-in-frequency assumption made by using the RBF kernel over frequency. Furthermore, the standard grid based approach to testing has difficulty diagnosing noise induced hearing loss because a frequency–intensity grid must be very fine to ensure that a notch is detected.

We cast the detection of noise-induced hearing loss as an active model selection problem. We will describe two Gaussian process models of audiometric functions: a baseline model of normal human hearing, and a model reflecting NIHL. We then use the BAMS framework introduced above to, as rapidly as possible for a given patient, determine which model best describes his or her hearing.

**Normal-patient model**. To model a healthy patient's audiometric function, we use the model described in [25]. The GP prior proposed in that work combines a constant prior mean $\mu_{\text{healthy}} = c$ (modeling a frequency-independent natural threshold) with a kernel taken to be the sum of two components: a linear covariance in intensity and a squared-exponential covariance in frequency. Let $[i, \phi]$ represent a tone stimulus, with $i$ representing its intensity and $\phi$ its frequency. We define:

$$K\big([i,\phi],[i',\phi']\big) = \alpha i i' + \beta \exp\big(-\tfrac{1}{2\ell^2}|\phi-\phi'|^2\big), \tag{4.11}$$

where $\alpha, \beta > 0$ weight each component and $\ell > 0$ is a length scale of frequency-dependent random deviations from a constant hearing threshold. This kernel encodes two fundamental properties of human audiologic response. First, hearing

is monotonic in intensity. The linear contribution $\alpha ii'$ ensures that the posterior probability of detecting a fixed frequency will be monotonically increasing after conditioning on a few tones. Second, human hearing ability is locally smooth in frequency, because nearby locations in the cochlea are mechanically coupled. The combination of $\mu_{\text{healthy}}$ with $K$ specifies our healthy model $\mathcal{M}_{\text{healthy}}$, with parameters $\theta_{\text{healthy}} = [c, \alpha, \beta, \ell]^\top$.

**Noise-induced hearing loss model**. We extend the model above to create a second GP model reflecting a localized, notch-shaped hearing deficit characteristic of NIHL. We create a novel, flexible prior mean function for this purpose, the parameters of which specify the exact nature of the hearing loss. Our proposed notch mean is:

$$\mu_{\text{NIHL}}(i, \phi) = c - d\mathcal{N}'(\phi; \nu, w^2), \tag{4.12}$$

where $\mathcal{N}'(\phi; \nu, w)$ denotes the unnormalized normal probability density function with mean $\nu$ and standard deviation $w$, which we scale by a depth parameter $d > 0$ to reflect the prominence of the hearing loss. This contribution results in a localized subtractive notch feature with tunable center, width, and height. We retain a constant offset $c$ to revert to the normal-hearing model outside the vicinity of the localized hearing deficit. Note that we completely model the effect of NIHL on patient responses with this mean notch function; the kernel $K$ above remains appropriate. The combination of $\mu_{\text{NIHL}}$ with $K$ specifies our NIHL model $\mathcal{M}_{\text{NIHL}}$ with, in addition to the parameters of our healthy model, the additional parameters $\theta_{\text{NIHL}} = [\nu, w, d]^\top$.

## 4.6 Results

To test BAMS on our NIHL detection task, we evaluate our algorithm using audiometric data, comparing to several baselines. From the results of the clinical trial discussed in Chapter 3, we have examples of high-fidelity audiometric functions inferred for several human patients. We may use these to simulate audiometric examinations of healthy patients using different methods to select tone presentations. We simulate patients with NIHL by adjusting ground truth inferred from nine healthy patients with in-model samples from our notch mean prior. Recall that high-resolution audiogram data is extremely scarce.

We first took a thorough pure-tone audiometric test of each of nine patients from our trial with normal hearing using 100 samples selected using the algorithm in Chapter 3 on the domain $\mathcal{X} = [250, 8000]\,\text{Hz} \times [-10, 80]\,\text{dB HL}$,[3] typical ranges for audiometric testing [41]. We inferred the audiometric function over the entire domain from the measured responses, using the healthy-patient GP model $\mathcal{M}_{\text{healthy}}$ with parameters learned via MLE−II inference. The observation model was $p(y = 1 \mid f) = \Phi(f)$, where $\Phi$ is the standard normal CDF, and approximate GP inference was performed via a Laplace approximation. We then used the approximate GP posterior $p(f \mid \mathcal{D}, \hat{\theta}, \mathcal{M}_{\text{healthy}})$ for this patient as ground-truth for simulating a healthy patient's responses. The posterior probability of tone detection learned from one patient is shown in the background of Figure 4.2. We simulated a healthy patient's response to a given query tone $\mathbf{m}\mathrm{x}^* = [i^*, \phi^*]$ by sampling a conditionally independent Bernoulli random variable with parameter $p(y^* = 1 \mid \mathbf{m}\mathrm{x}^*, \mathcal{D}, \hat{\theta}, \mathcal{M}_{\text{healthy}})$.

We simulated a patient with NIHL by then drawing notch parameters (the

[3] Inference was done in log-frequency domain.

parameters of (4.12)) from an expert-informed prior, adding the corresponding notch to the learned healthy ground-truth latent mean, recomputing the detection probabilities, and proceeding as above. Example NIHL ground-truth detection probabilities generated in this manner are depicted in the background of Figure 4.2.

### 4.6.1 Diagnosing NIHL

To test our active model-selection approach to diagnosing NIHL, we simulated a series of audiometric tests, selecting tones using three alternatives: BAMS, the active learning audiometric algorithm presented in Chapter 3, and random sampling.[4] Each algorithm shared a candidate set of 10 000 quasirandom tones $X^*$ generated using a scrambled Halton set so as to densely cover the two-dimensional search space. We use data from nine healthy patients and simulate a total of 27 patients exhibiting a range of NIHL presentations, using independent draws from our notch mean prior in the latter case. For each audiometric test simulation, we initialized with five random tones, then allowed each algorithm to actively select a maximum of 25 additional tones, a very small fraction of the hundreds typically used in a regular audiometric test. We repeated this procedure for each of our nine healthy patients using the normal-patient ground-truth model. We further simulated, for each patient, three separate presentations of NIHL as described above. We plot the posterior probability of the correct model after each iteration for each method in Figure 4.3.

In all runs with both ground-truth models, BAMS was able to rapidly achieve

[4]We also compared with uncertainty sampling and query by committee (QBC); the performance was comparable to random sampling and is omitted for clarity.

greater than 99% confidence in the correct model without expending the entire budget. Although all methods correctly inferred high healthy posterior probability for the healthy patient, BAMS was more confident. For the NIHL patients, neither baseline inferred the correct model, whereas BAMS rarely required more than 15 actively chosen samples to confidently make the correct diagnosis. Note that, when BAMS was used on NIHL patients, there was often an initial period during which the healthy model was favored, followed by a rapid shift towards the correct model. This is because our method penalizes the increased complexity of the notch model until sufficient evidence for a notch is acquired.

Figure 4.2 shows the samples selected by BAMS for typical healthy and NIHL patients. The fundamental strategy employed by BAMS in this application is logical: it samples in a row of relatively high-intensity tones. The intuition for this design is that failure to recognize a normally heard, high-intensity sound is strong evidence of a notch deficit. Once the notch has been found (Figure 4.2), BAMS continues to sample within the notch to confirm its existence and rule out the possibility of the miss (tone not heard) being due to the stochasticity of the process. Once satisfied, the BAMS approach then samples on the periphery of the notch to further solidify its belief.

The BAMS algorithm sequentially makes observations where the healthy and NIHL model disagree the most, typically in the top-center of the MAP notch location. The exact intensity at which BAMS samples is determined by the prior over the notch-depth parameter $d$. When we changed the notch depth prior to support shallower or deeper notches (data not shown), BAMS sampled at lower or higher intensities, respectively, to continue to maximize model disagreement. Similarly, the spacing between samples is controlled by the prior over the notch-

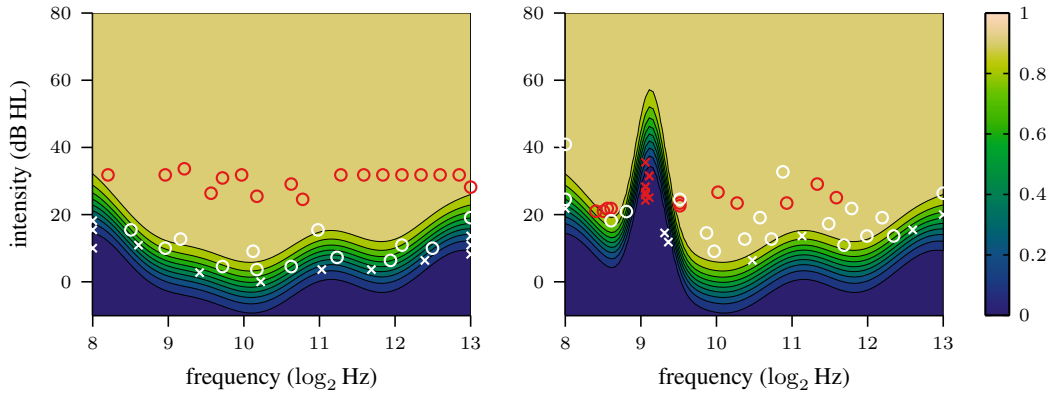


Figure 4.2: Samples selected by BAMS (red) and the method from Chapter 3(white) when run on the normal-hearing ground truth and the NIHL model ground truth. Contours denote probability of detection at 10% intervals. Circles indicate presentations that were heard by the simulated patient; exes indicate presentations that were not heard by the simulated patient. **Left:** Normal hearing model ground truth. **Right:** Notch model ground truth.

width parameter $w$.

Finally, it is worth emphasizing the stark difference between the sampling pattern of BAMS and the audiometric test from Chapter 3; see Figure 4.2. Indeed, when the goal is learning the patient's audiometric function, the audiometric testing algorithm proposed in that work typically has a very good estimate after 20 samples. However, when using BAMS, the primary goal is to detect or rule out NIHL. As a result, the samples selected by BAMS reveal little about the nuances of the patient's audiometric function, while being highly informative about the correct model to explain the data. This is precisely the tradeoff one seeks in a large-scale diagnostic setting, highlighting the critical importance of focusing on the model-selection problem directly.

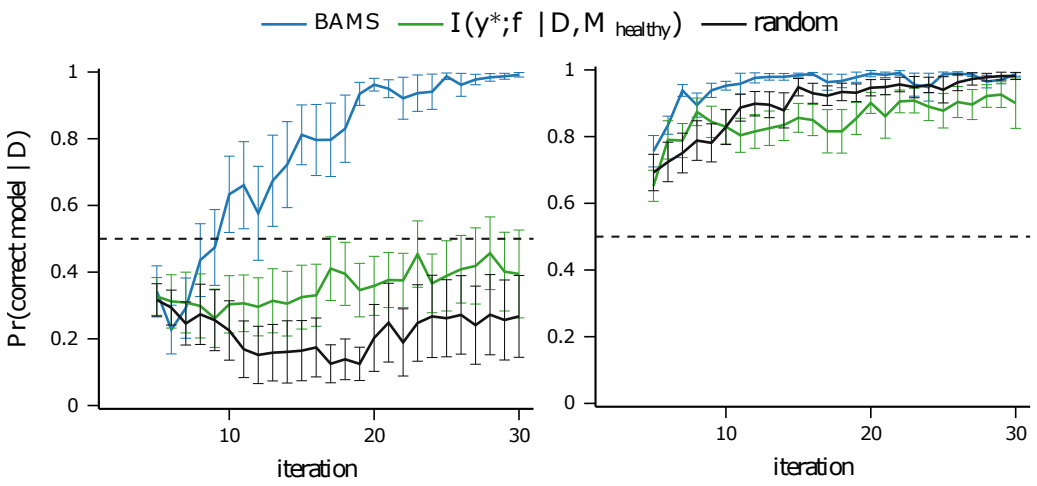


Figure 4.3: Posterior probability of the correct model as a function of iteration number. **Left:** Notch model ground truth. **Right:** Normal hearing model ground truth.

## 4.7 Discussion

We have now seen a novel information-theoretic approach for active model selection, *Bayesian active model selection*, and successfully applied it to rapid screening for noise-induced hearing loss. Our method for active model selection does not require model retraining to evaluate candidate points, making it more feasible than previous approaches. Further, we provided an effective and efficient analytic approximation to our criterion that can be used for automatically learning the model class of Gaussian processes with arbitrary observation likelihoods, a rich and commonly used class of potential models.

**Impact.** Beyond audiometric testing, one of the key methodological contributions of this paper is an extension of covariance function selection techniques based on Bayesian model selection to the active learning setting, and in particular the small data setting. Indeed, one of the broader take-aways from this work is the simple example in Figure 4.3: sufficiently different covariance functions can

often be distinguished almost perfectly with a very small number of examples. On the application side, it is our hope that this work will eventually be adopted and supplant existing techniques used for noise-induced hearing loss screening. It is our understanding that standard practice for this problem is to give a subject a full audiometric test; however, as we've demonstrated, this can potentially be quite slow or even fail to discover particularly sharp notches entirely.

**Future work.** As a consequence of this insight, one of the most obvious future directions for this work is to use these techniques for more popular applications of Gaussian processes, and in particular Bayesian optimization. The ability to choose significantly more informative priors with a small number of training examples is an interesting prospect in the Bayesian optimization framework in particular. Indeeed in the next chapter, we will focus on one specific setting where covariance function learning in the limited data regime dramatically improves convergence.

CHAPTER 5

# DISCOVERING AND EPXLOITING ADDITIVE STRUCTURE FOR BAYESIAN OPTIMIZATION

## 5.1 Introduction

In the previous chapter, we developed a technique that extends Bayesian model selection to the active learning setting and allows for automatically selecting an appropriate covariance function for the modeling task at hand, often after only a handful of examples. While this approach is extremely effective when the space of models to choose from is small, it does have the drawback of needing to train a Gaussian process on all observed data with each candidate covariance function. While this is not problematic when the space of candidate models is reasonable, this becomes untenable if the candidate model space becomes gigantic, even when using scalable GP techniques.

In this chapter, we turn to a second application of model selection with limited data, but one that must select from a *combinatorially large* set of covariance functions. In particular, we will investigate whether model selection during *Bayesian optimization* can be used to more rapidly optimize the target blackbox function than simply using a standard RBF or Matern kernel.

It is worth noting that a typical BayesOpt algorithm has only two key factors that must be selected by the user: the acquisition function—how new queries are selected given the posterior belief—and the choice of prior. Whereas the former has been studied extensively, and many reasonable options exist, the choice of prior has received very little attention. Almost all off-the-shelf BayesOpt solvers

use general-purpose kernels, and do little in the way of discovering and exploiting structure that may underlie a particular objective function.

With poor or overly general choices of the kernel, BayesOpt may converge very slowly on complex functions, especially in moderate-to-high dimension. Hence, exploiting structure can have a substantial impact on optimization performance. For example, [90] argue that for hyperparameter optimization a Matérn-5/2 kernel provides better results in general than the squared exponential kernel, because the latter models unrealistically smooth functions. Choosing (or learning) a kernel that exploits low-dimensional structure in the function can significantly improve performance when confronted with 10s or 100s of parameters [102, 26].

In this chapter, we will seek to apply model selection to discover and exploit such *additive structure:* cases where the setting of some variables in the design space does not affect the optimal setting of others. For example, in hyperparameter tuning, some groups of parameters may have well-known interactions (e.g., learning rate and momentum), whereas others do not strongly interact (e.g., momentum and regularization weight). Intuitively, if different groups of parameters interact only additively, then they can be optimized independently. This intuition has been formalized many times in the context of kernels [3], generalized additive models [35], and various extensions [101]. In the context of Bayesian optimization specifically, [45] recently showed that knowing the additive structure of a function ahead of time gives *exponential* reductions in sample complexity.

However, it is challenging in general to reason about the structure of arbitrary black-box functions *a priori.* The core contribution of our work is an efficient mechanism for discovering the underlying structure of the function *while*

BayesOpt is executed. Although in this chapter we focus on additive structure, we believe this technique could be used more broadly in a wide variety of settings.

Rather than fixing a model *a priori*, we make use Bayesian model selection to, at each iteration, sample an additive structure consistent with the data observed so far. We deal with the large number of possible additive structures by using a Metropolis–Hastings scheme to sample from the model posterior, and demonstrate that this mechanism quickly samples additive structures that explain the data well.

We evaluate our method both in-model and on several benchmark optimization problems with varying additive structure. The experiments validate empirically that our approach significantly outperforms both standard BayesOpt and the random-bag-of-models exploration scheme introduced by [45]. On synthetic test functions, we show that our approach discovers the correct structure even for functions in a moderately large number of dimensions and is substantially faster than existing structure-discovery techniques.

Finally, we evaluate the efficacy of our proposed algorithm on two real-world applications: a matrix completion task and an astrophysics simulation experiment to estimate physical constants to high accuracy. While these real-world settings were not known to have any form of additive structure *a priori*, our method nevertheless converges faster and finds better optimal solutions than prior work. As considering additive structure is a strict generalization of using the fully dependent structure, it is our hope that the methods discussed in this Chapter can be broadly applied: in all of our testing, we did not notice a setting where *attempting* to discover additive structure was *detrimental* to performance.

## 5.2 Related Work

The idea that modeling functions can be done more efficiently by exploiting underlying additive structure is well known in general [3]. Generalized additive models (GAMs) are linear models that explicitly model a response variable as a linear combination of univariate functions [35]. These results have been extended to the setting where some component functions depend on more than one input variable, allowing for general additive structure [101]. In the context of Gaussian process regression in machine learning, several papers exist that exploit general forms of additive structure in the literature. [22] introduced an additional generaliztion of GAMs that allows for "higher-order" additive kernels (GAMs corresponding to first-order additive kernels).

[45] demonstrated that Bayesian optimization using additive Gaussian processes achieves lower regret than fully dependent models. They showed that Bayesian optimization using a prior that exploits additive structure in the objective function has *theoretically* significantly lower sample complexity than with a kernel that does not exploit this structure. In their paper, the authors proposed evaluating a number of randomly selected additive structures—which we will call the *bag-of-models* approach—and choosing the structure that explains any observed data the best. The primary goal of our work is to show that this random search is inefficient and can be improved dramatically by performing an *informed* search in model space in tandem with the optimization procedure.

The problem of choosing covariance functions that result in better models than basic kernels has been well studied in general. One approach to this is to compose simple kernels (for example, the RBF kernel, the linear kernel, etc) using

simple operations like addition and multiplication [74]. [19] proposed to search over possible compositions by constructing a *grammar* of possible kernel compositions, starting with basic base kernels and producing more complex kernels via composition. This tree is then greedily searched over to construct a kernel with high model evidence. [59] extended this idea, replacing the greedy search with Bayesian optimization in model space, defining a novel "kernel kernel" to reason about the similarity between data explanations offered by different kernels and speed up the search. Rather than constructing complex models from simple ones, [108] took a different approach, and introduced a highly flexible kernel by using Bochner's theorem to write any stationary kernel as the Fourier transform of a finite measure, and parameterizing a class of stationary kernels by using a Gaussian mixture as the spectral density. Similar to our setting, [24] considered actively discovering the structure of a function by sequentially querying as few points as possible. However, the techniques in that paper are not used for optimization.

Choosing generally more informative priors in Bayesian optimization has received some attention as well. For example, [95] considered using information gained from functions related to the objective to guide optimization. This occurs, for example, in hyperparameter tuning when hyperparameters on subsampled data can be used to inform about the validation error on the full dataset.

## 5.3 Model selection via MCMC

The number of possible additive decomposition models in $d$ dimensions is given by the $d$th Bell number [105], which satisfy the following recurrence relation (for

$d > 1$) :

$$B_d = \sum_{k=0}^{d-1} \binom{d-1}{k} B_k, B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k,$$

with $B_0 = B_1 = 1$. This grows super-exponentially, and for $d = 10$ already reaches $B_{10} = 115\,975$. This makes computing the model posterior in eq. (2.13) prohibitively expensive, as it involves conditioning a Gaussian process with each of these models to compute the model evidences. However, computing a small (i.e., not super-exponential) number of model evidences is tractable. A natural alternative to computing the model posterior is to sample from it using Metropolis–Hastings, which requires only one additional model evidence computation per sample.

**Proposal distribution.** As we focus on additive structure throughout this section, a model is uniquely defined by its partitioning of the underlying dimensions, and with a slight abuse of notation we refer to the partitioning and the resulting model both as $\mathcal{M}$. The primary component of Metropolis–Hastings that needs to be specified is the *proposal distribution* $g(\mathcal{M}' \mid \mathcal{M})$. Given an additive structure partitioning $\mathcal{M}$, there are two natural operations that can be performed.

First, an existing element of the partition can be split in two. For example, if $[1, 2, 3] \in \mathcal{M}$, we can form $[1][2, 3], [2][1, 3], [3][1, 2]$ by splitting. In general, a component of size $k$ can be split in $2^{k-1}$ different ways.

Second, two existing elements of the partition may be merged. For example, $[1, 4]$ and $[2, 3]$ can be merged to form $[1, 2, 3, 4]$. In general, a partition with $k$ components has $\binom{k}{2}$ possible merges.

With these operations in mind, we construct a proposal distribution as illus-

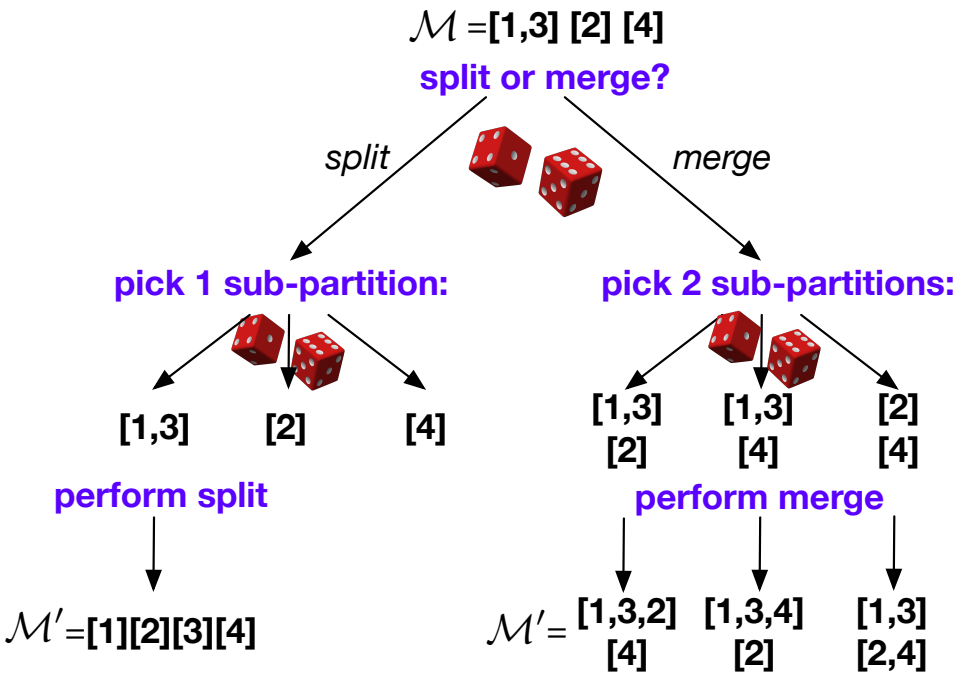


Figure 5.1: An illustration of the proposal distribution $g(\mathcal{M}' \mid \mathcal{M})$ on a simple input model with three sub-partitions, $\mathcal{M} = [1,3][2][4]$. Splits with dice represent choices performed uniformly at random (without replacement). See text for details.

trated in Figure 5.1. Since there may be more splits than merges or vice versa depending on the current state of the Markov chain, we first choose whether to split or merge, each with $50\%$ probability. Next, if we split (left sub-tree), we choose a component of the existing partition $\mathcal{M}$ uniformly at random and split it in two (again uniformly at random). If we instead merge (right sub-tree), we choose two components of $M$ uniformly without replacement and merge them. Given this proposal mechanism, sampling from the proposal distribution $g(\mathcal{M}' \mid \mathcal{M})$ is straightforward and efficient.

## 5.4 Additive structure discovery

The goal of our work is to discover the additive model structure underlying the objective function $f(\mathbf{x})$, while simultaneously exploiting it for BayesOpt.

At each iteration $i$ of BayesOpt, we will have collected some dataset of function evaluations $\mathcal{D}_i = \{X_i, \mathbf{y}_i\}$. BayesOpt, as described in section 2.3, uses this data to update the posterior $p(f \mid \mathcal{D}_i)$ and then to identify the new point $\mathbf{x}^*$, which maximizes EI($\mathbf{x}^*$) and for which $f(\mathbf{x}^*)$ should be evaluated next.

The expected improvement EI($\mathbf{x}^*$) is a function of $p(f(\mathbf{x}^*) \mid \mathcal{D}, \mathbf{x}^*)$. As we are considering multiple models, each providing us with a different posterior over $f$, we approximately marginalize out the model. We sample $k$ models $\mathcal{M}_1, \ldots, \mathcal{M}_k$ from $p(\mathcal{M} \mid \mathcal{D}_i)$ to obtain

$$p(f(\mathbf{x}^*) \mid \mathcal{D}, \mathbf{x}^*) \approx \frac{1}{k} \sum_{j=1}^{k} p(f(\mathbf{x}^*) \mid \mathcal{D}, \mathbf{x}^*, \mathcal{M}_j). \tag{5.1}$$

**MCMC.** To obtain these samples, we use the Metropois-Hastings algorithm with the proposal distribution $g(\mathcal{M} \mid \mathcal{M}')$. First observe that the model posterior is proportional to the model evidence $p(\mathbf{y} \mid X, \mathcal{M})$, by eq. (2.13), which can be efficiently evaluated for single models. We can therefore sample $k$ models as follows: Given the current model $\mathcal{M}_j$ (initializing $\mathcal{M}_0$ to the final model found in the previous iteration), we sample a proposed model $\mathcal{M}'$ from the proposal distribution $g(\mathcal{M}' \mid \mathcal{M}_j)$. Next, we compute the model evidence for $\mathcal{M}'$ and use this to compute the Metropolis-Hastings acceptance probability:

$$A(\mathcal{M}' \mid \mathcal{M}_j) = \min\left(1, \frac{p(\mathbf{y}_i \mid X_i, \mathcal{M}')g(\mathcal{M}_j \mid \mathcal{M}')}{p(\mathbf{y}_i \mid X_i, \mathcal{M}_j)g(\mathcal{M}' \mid \mathcal{M}_j)}\right).$$

Finally, we update the current state to $\mathcal{M}'$ with probability $A(\mathcal{M}' \mid \mathcal{M}_i)$.

**Candidate selection.** As pointed out at the end of section 2.3, finding the point $\mathbf{x}_i^*$ to maximize $\text{EI}(\mathbf{x}^*)$ has exponential sample complexity with the number of dimensions and can be slow. This originates from the fact that the acquisition function is evaluated on a fine grid in the $d$ dimensional space, or is optimized using methods that scale poorly with dimensionality (for example, gradient descent with random restarts or DIRECT). In the presence of additive structure, this search can be decomposed and performed for each component individually—leading to exponential speed-ups. To do this, we start with some initial $\mathbf{x}_i^*$ and iteratively optimize EI in each component of the partition. For example, if the first component of the partition is [1,3,5], we first optimize the first, third, and fifth dimensions of $\mathbf{x}_i^*$, holding the other dimensions fixed. If the next component is [4], we optimize EI by varying the 4th dimension (holding other dimensions fixed), and so on.

As each model $\mathcal{M}_i$ has its own additive structure, we identify the best $\mathbf{x}_i^*$ for each of the $k$ models,

$$\mathbf{x}_i^* = \arg\max_{\mathbf{x}} \text{EI}(\mathbf{x}_i \mid \mathcal{M}_i). \tag{5.2}$$

We consider the resulting points $\mathbf{x}_1, \ldots \mathbf{x}_k$ as candidate points and set $\mathbf{x}^* = \mathbf{x}_i$ for the particular $i$ that maximizes the marginalized acquisition function $\text{EI}(\mathbf{x}_i^*)$ using eq. (5.1).

## 5.5 Results

For all the experiments, we used the Python GP toolbox GPy [31].

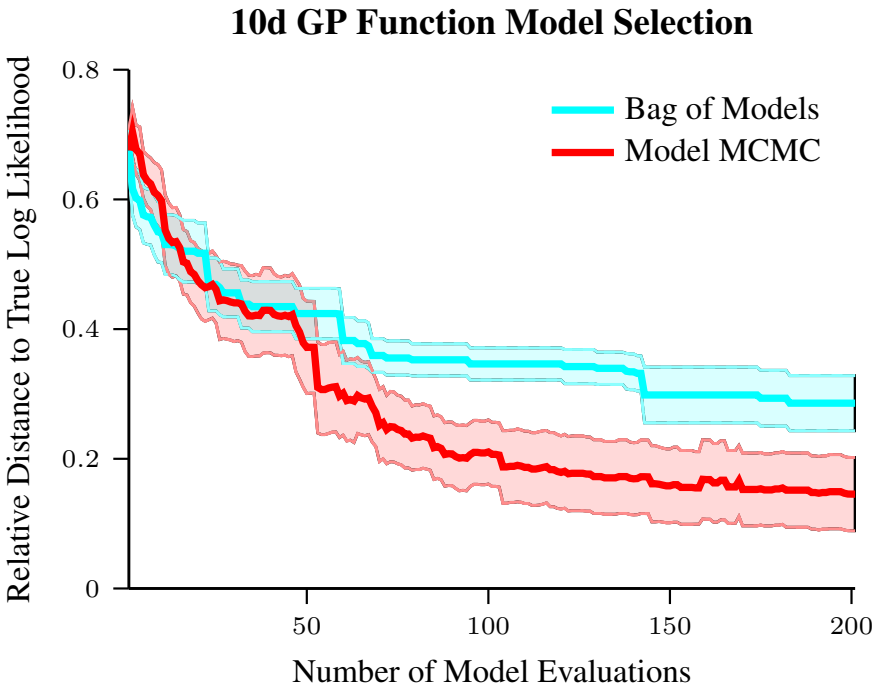


Figure 5.2: Plot of the relative distance to the true loglikelihood as a function of the number of model evaluations (see text). For MCMC, this means the number of samples drawn in the Markov chain. For bag of models, this means the number of models in the selected bag. Error bars are standard error averaged over 10 runs.

### 5.5.1 Model Selection

Here we demonstrate the effectiveness of MCMC search in model space. We sample a random partition $P$ of $\{1, \ldots, d\}$ and sample observations from a Gaussian process with additive structure corresponding to $P$. In particular, we take $N = 50$ points from a scrambled Halton sequence on $[0, 1]^d$, and sample corresponding function values $f(\mathbf{x})$ from the Gaussian process. We then perform MCMC model search using these $N$ observations. The initial kernel is the fully dependent kernel. Figure 5.2 shows plots of the Markov chain length versus model evidence achieved with that chain length for $d = 10$. We compare with the bag-of-models (BOM) proposed by [45]. For each method we plot, as a function of the number of models evaluated, the *relative distance* to the true model.

This is given by $1 - \frac{\log p(\mathbf{y}|X,\mathcal{M}_i,\theta_i)}{\log p(\mathbf{y}|X,\mathcal{M}^*,\theta^*)}$, where $\mathcal{M}^*$ is the true model and $\mathcal{M}_i$ is the model evaluated at iteration $i$.

MCMC consistently finds additive decompositions within 15–17% of the ground-truth log likelihood. In 8 out of 10 runs, MCMC samples the ground-truth model exactly. BOM consistently fails to find models with a relative log likelihood distance of less than $30\%$. This experiment demonstrates that MCMC is an effective technique for exploring the space of additive structures, at least when the underlying function is well modeled by a Gaussian process. Specifically, we can expect it to outperform the BOM when used as a subroutine in Bayesian optimization when additive structure is critical to the optimization task.

### 5.5.2 Optimization

We next incorporate MCMC model search into the BayesOpt framework. In all experiments below, we sample $k = 50$ models using the MCMC techniques discussed above at each iteration. For the Bag of Models (BOM) baseline, we use a bag of $50$ models. Thus, the MCMC approach and the bag of models approach perform the same number of MLE optimizations of the log likelihood $\log p(\mathbf{y} \mid X, \mathcal{M}_i, \theta_i)$. Instead of performing *burn-in*—discarding initial low-likelihood samples—we initialize the Markov chain at each iteration with the final model sampled at the previous iteration. Therefore, the MCMC and BOM approaches add virtually identical amounts of overhead to the baseline BayesOpt algorithm, and the wallclock time required for each iteration is essentially identical for both methods.

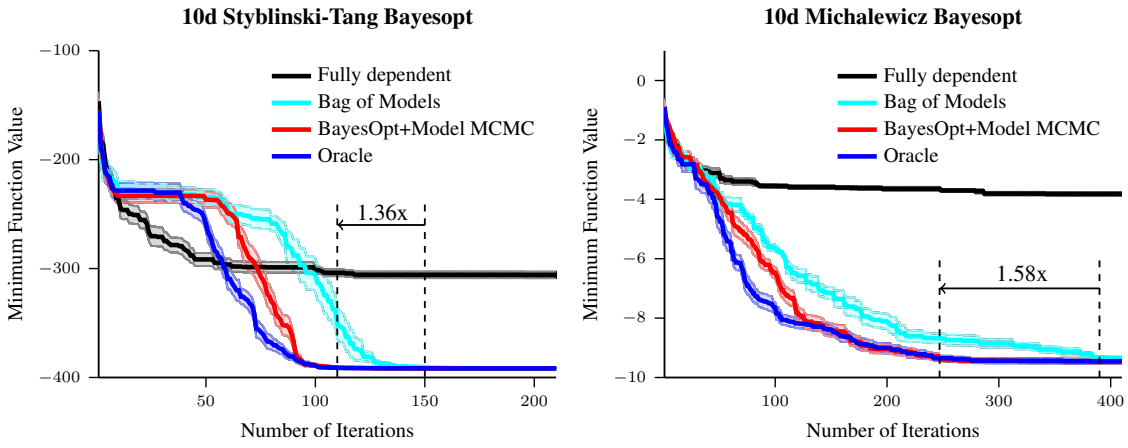


Figure 5.3: Optimizing the 10d Styblinksi-Tang and Michalewicz functions. Shaded error bars are 2 standard errors over 10 runs. BayesOpt+Model MCMC converges faster than BOM in both cases.

We do note that the amount of overhead added by performing model selection is not inconsequential for cheap functions $f$. Indeed, considering $k$ models at each iteration results in roughly a factor $k$ increase in the overhead added by BayesOpt. However, in many real world settings (like the cosmological constants experiment and the matrix factorization experiment below), the overhead of running BayesOpt is insignificant compared to evaluating the objective function $f$) as long as only a relatively small number of optimization iterations are run.

In all experiments, the squared exponential (SE) kernel is used as the base kernel.

### 5.5.3 Optimization of Benchmark Functions

We first consider two standard optimization benchmark functions that have additive structure: the Styblinski–Tang function, and the Michalewicz function. The

$d$-dimensional Styblinski–Tang function is defined as

$$\text{Stybtang}(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{d} x_i^4 - 16x_i^2 + 5x_i.$$

The global minimum is approximately $-39.166d$ and is attained at point $\mathbf{x}^* \approx (-2.9, \ldots, -2.9)$. We restrict the domain of the function to $[-4, 4]^d$ for the optimization.

The $d$-dimensional Michalewicz function is defined as

$$\text{Michalewicz}(\mathbf{x}) = -\sum_{i=1}^{d} \sin(x_i) \sin^{2m}\left(\frac{ix_i}{\pi}\right).$$

Here, $m$ controls the steepness of valleys and ridges. For this experiment, we set $m = 10$. For $d = 10$, the global minimum is approximately $-9.66$ over the domain $[0, \pi]^d$.

In addition to these two functions, we extend the Styblinski–Tang function, which fully additively decomposes, to a *transformed* Styblinski–Tang function: We sample a random partition $P$ and for each part $i$ of $P$, we sample a random orthonormal matrix $Q_i$ over the dimensions of part $i$. If $Q$ is the block diagonal matrix formed by placing each $Q_i$ on a diagonal, then Stybtang$(Qx)$ is no longer fully additive, but instead is additive across the components of $P$. We use this to investigate performance when the true function is not fully additive across individual dimensions.

We compare against three baselines. The fully dependent model uses a single $d$-dimensional kernel as the model. This is the approach used by popular BayesOpt packages such as Spearmint [90]. The oracle model uses a sum of squared-exponential kernels whose parts capture the true additive structure of the underlying function. For all experiments, we expect the oracle model to outperform all other methods since it has prior knowledge of the true function

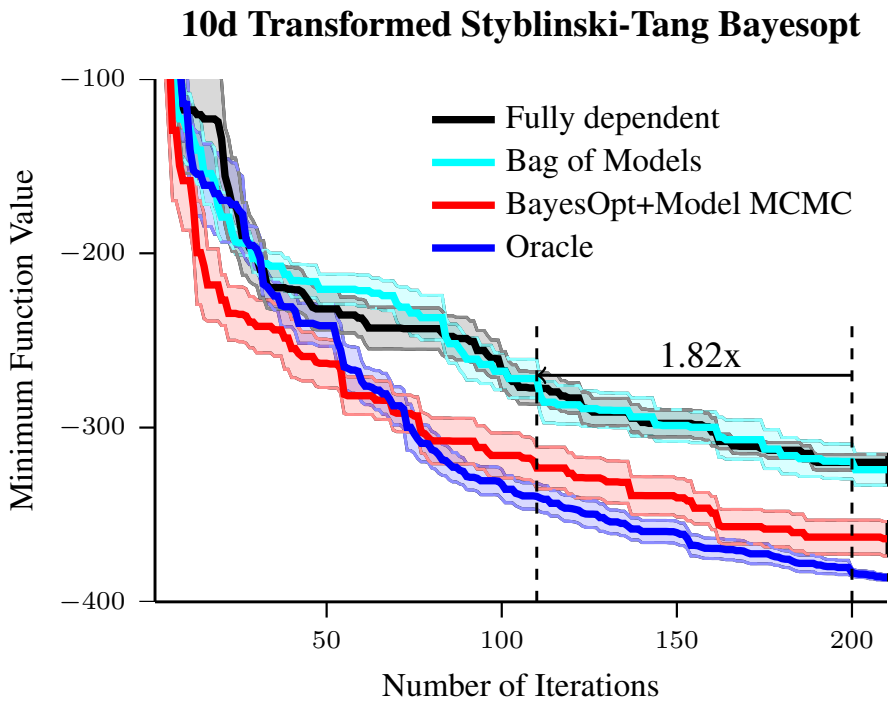


Figure 5.4: Optimizing the 10d transformed Styblinksi-Tang function. Shaded error bars are 2 standard errors over 10 runs. BayesOpt+Model MCMC converges to the same final solution as BOM much faster, and significantly outperforms it in terms of final objective value.

structure. Our third baseline is BOM [45]. In all experiments, BayesOpt+Model MCMC significantly outperforms the non-oracle approaches, both in terms of convergence speed and ultimate objective value.

Figure 5.3 left shows the plot of optimizing the Styblinski–Tang function. Each curve shows the mean cumulative minimum function evaluation up to the current iteration across 10 runs, while the shaded region shows the standard error. It can be seen that our method attains the true global minimum as quickly as the oracle model, while outperforming BOM. The fully dependent model does not converge within 200 iterations.

Figure 5.3 right shows the plot of optimizing the Michalewicz function. Since the function has many steep valleys, it is much more difficult to model with a

Gaussian process than the Styblinski–Tang function. Nevertheless, both the oracle model and our method converge to the global optimum within 300 iterations while BOM has not yet converged after 400 iterations. Again, the fully dependent model shows no sign of convergence.

In the transformed Styblinski-Tang experiment (Figure 5.4), the more difficult model selection problem poses a significant challenge to the BOM method, which does not significantly outperform the baseline fully dependent model. In contrast, Model MCMC successfully discovers appropriate additive structure and performs significantly better. Of particular interest is the first 50 iterations where Model MCMC outperforms the baseline. This happens in cases where the ground truth model has large non-additive components, which makes the EI optimization problem significantly more difficult for the oracle method. The Model MCMC method averages over many additive structures which, while incorrect, have much simpler structure and for which EI can be more efficiently optimized.

### 5.5.4 Determining Cosmological Parameters

In this section, we test our method on the task of determining the values of various cosmological constants (e.g., Hubble's constant, the density of baryonic matter in the universe, etc). These cosmological constants are values that are required in standard physical models of the universe, but their exact values must be determined experimentally. To do this, scientists can run simulations with various settings of these cosmological constants and evaluate how well these simulations model experimentally observed data.

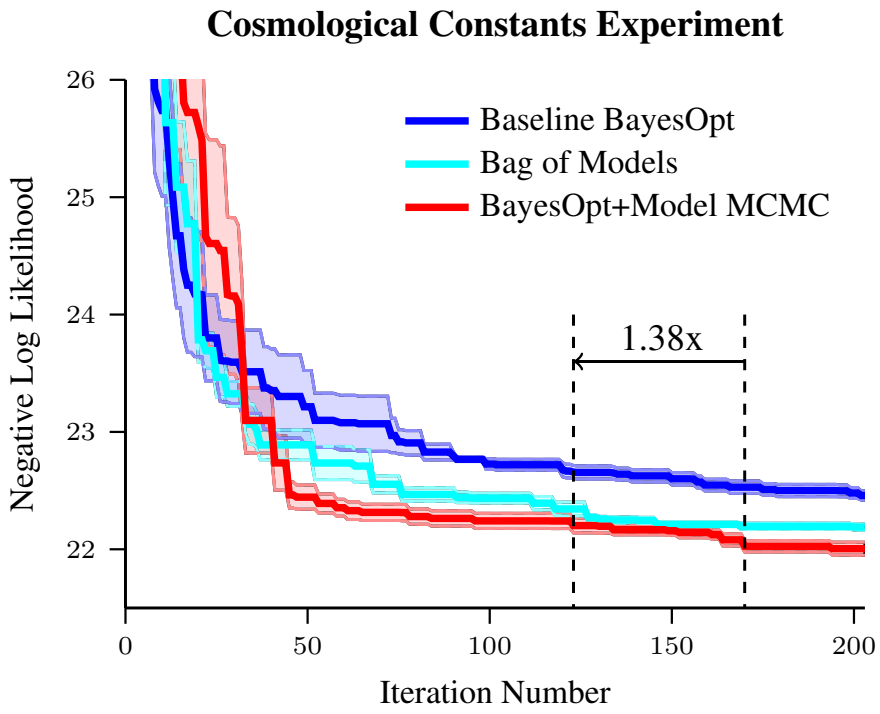


Figure 5.5: Setting values of various cosmological constants to match experimental data using BayesOpt. Shaded regions correspond to 2 standard errors over 10 runs. BayesOpt+Model MCMC matches BOM 38% faster, and then outperforms it.

We use software released by NASA[1] that, given settings of these cosmological constants, computes via simulation the likelihood of a set of experimental data released by the Sloan Digital Sky Survey. As in [45], we tune the 9 parameters that this software takes as input and produces the negative log likelihood. However, unlike [45], we do not introduce placeholder dimensions that have no effect on the objective function value.

To set ranges for each of the parameters, we take the values set in a parameter file shipped with the software, and optimize over a range of $75\% - 125\%$ of this default value for each parameter. The results of this experiment are in Figure 5.5.

The parameter values shipped with the software achieve an objective func-

[1]https://lambda.gsfc.nasa.gov/toolbox/lrgdr/

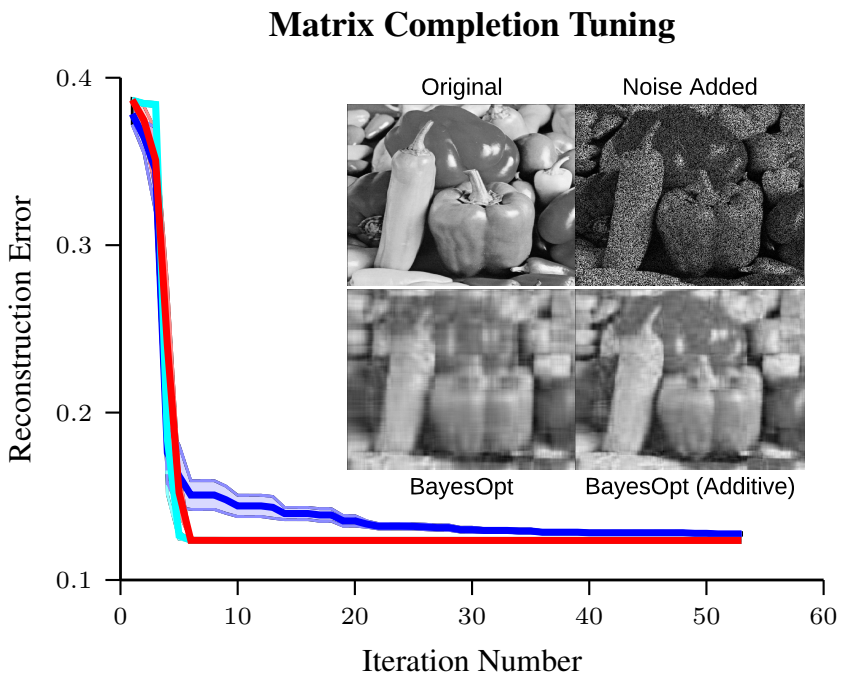


Figure 5.6: Optimizing the reconstruction error of images with matrix completion. Corner: Example of image reconstruction on the peppers image using hyperparameters found by both the baseline and by BayesOpt with additive structure. The additive BayesOpt parameters produce a significantly sharper image.

tion value of 23.7. All three methods find statistically significantly better solutions by 200 iterations, with the BayesOpt+Model MCMC approach performing the best, consistently converging to an optimal negative log likelihood of 22.17. Furthermore, BayesOpt+Model MCMC converges much faster, achieving the global solution found by the bag of models method nearly 40% faster.

### 5.5.5 Matrix completion

Many of the experiments in [45] and our paper here have focused on functions with $10$ or more dimensions. However, there are also many machine learning algorithms with only a small number ($3$ or fewer) of hyperparameters. Although with these few dimensions, employing the machinery of MCMC to explore the

model space is not necessarily critical (indeed, BOM reduces to evaluating all possible additive decompositions and choosing the best), we seek to demonstrate two things: First, exploiting additive structure can still have a substantial impact on performance. Second, even in a scenario where BOM is optimal (i.e. the best additive structure is chosen each iteration), our sampling approach matches its performance.

We evaluate our method on the task of choosing hyperparameters for a matrix completion algorithm in [9]: a singular-value decomposition-based method for recovering a matrix with missing entries which can be used for image denoising, recommendation, and data interpolation, among other applications. We set the following hyperparameters using standard BayesOpt, our method, and BOM: the regularization trade-off $\tau$, the step-size $\delta$, and the noise-constraint $\varepsilon$. A plot of the average validation reconstruction error on an image reconstruction task as a function of the number of BayesOpt iterations is displayed in figure 5.6.

Both additive structure mechanisms significantly outperform the standard BayesOpt baseline, both in terms of convergence speed and in terms of final objective function value. To assess whether the hyperparameters made a difference in terms of image reconstruction quality, we use the hyperparameters found by each method to recover the benchmark "peppers" image [103] after noise has destroyed $50\%$ of the pixels. The corrupted image as well as the reconstructions are in figure 5.6. The image reconstructed using the additive Bayesian optimization methods is significantly sharper than the image reconstructed using the non-additive approach, with less blurring and much sharper edges.

## 5.6 Discussion

**Problem structure for Bayesian optimization.** It's important to emphasize that the choice of prior has rarely been considered for Bayesian optimization historically. Largely, this has been due to the fact that it is difficult for an algorithm designed to optimize arbitrary unknown functions to take advantage of problem structure. It would be difficult to *a priori* suppose that montonicity or periodicity exists in a blackbox function, and therefore implementations have largely defaulted to general purpose covariance functions that have universal approximation properties. This can result in very high sample complexities, but makes it more likely that Bayesian optimization will *eventually* succeed.

**Beyond additive structure.** The primary reason that the methods in this chapter are so effective is that additive structure is an extremely strong prior to impose on the blackbox function, particularly for optimization: if two variables of the optimization problem exist only in different additive components, then the covariance function only models functions in which the variables do not "interact" at all. In general, one could imagine the set of parameters as a graph, with an edge between two parameters if they exist in the same component. The standard RBF kernel would correspond here to the fully connected graph, while the fully additive model corresponds to the graph with no edges. In general, one could consider arbitrary graphs, with the reasonable expectation that fewer edges correspond to increasingly easier optimization tasks but increasingly more restrictive priors. It would be interesting to see what can be shown for more general structures. For example, is a function that decomposes in to two connected components with a single edge between them easier to optimize than the fully

connected structure?

**Conclusion.** In this chapter we introduced an integrated solution to discover and exploit additive structure while performing BayesOpt. Our algorithm is based on MCMC sampling of model candidates and in practice converges surprisingly fast for plausible and useful model decompositions. Given the drastic speed-ups that can be achieved through exploitation of additive structure, we hope that our algorithm will become a standard component of Bayesian optimization.

CHAPTER 6

# EXPLOITING PRODUCT STRUCTURE FOR SCALABLE GAUSSIAN PROCESSES

## 6.1 Introduction

In this chapter, we will look at how covariance function structure can be usefully exploited to dramatically increase the scalability of Gaussian processes in many common settings. Historically, one of the key limitations of Gaussian process regression has been the computational intractability of inference when dealing with more than a few thousand data points. This complexity stems from the need to solve linear systems and compute log determinants involving an $n \times n$ symmetric positive definite *covariance matrix* $K$. This task is commonly performed by computing the Cholesky decomposition of $K$ [73], incurring $\mathcal{O}\left(n^3\right)$ complexity. To reduce this complexity, *inducing point methods* make use of a small set of $m < n$ points to form a rank $m$ approximation of $K$ [70, 89, 36, 97]. Using the matrix inversion and determinant lemmas, inference can be performed in $\mathcal{O}\left(nm^2\right)$ time [89].

Recently, however, an alternative class of inference techniques for Gaussian processes have emerged based on iterative numerical linear algebra techniques [113, 17]. Rather than explicitly decomposing the full covariance matrix, these methods leverage Krylov subspace methods [28] to perform linear solves and log determinants using only matrix-vector multiples (MVMs) with the covariance matrix. Letting $\mu(K)$ denote the time complexity of computing $K\mathbf{v}$ given a vector $\mathbf{v}$, these methods provide excellent approximations to linear solves and

log determinants in $\mathcal{O}\left(k\mu(K)\right)$ time, where $k$ is typically some small constant.[1] This approach has led to scalable GP methods that differ radically from previous approaches – the goal shifts from computing efficient Cholesky decompositions to computing efficient MVMs. Structured kernel interpolation (SKI) [113] is a recently proposed inducing point method that, given a regular grid of $m$ inducing points, allows for MVMs to be performed in an impressive $\mathcal{O}\left(n + m \log m\right)$ time.

These MVM approaches have two fundamental drawbacks. First, in order for SKI to take advantage of fast MVMs, the number of inducing points $m$ grows exponentially with the dimensionality of the inputs, limiting the applicability of SKI to problems with fewer than $5$ input dimensions.

Second, the computational benefits of iterative MVM inference methods come at the cost of reduced modularity. If all we know about a kernel is that it decomposes as $K = K_1 \circ K_2$, it is not obvious how to efficiently perform MVMs with $K$, even if we have access to fast MVMs with both $K_1$ and $K_2$. As we've seen throughout this thesis, the ability to compose covariance functions using addition and multiplication is a critical feature of Gaussian processes, as it allows practioniers to exploit problem specific structure for sample efficient learning.

In order for MVM inference to truly support the expressive expressive modeling choices offered by composing covariance functions, we should be able to perform inference equipped with nothing but the ability to perform MVMs with $K$. One of the most common kernel compositions is the element-wise product of kernels. This composition can encode different functional properties for each input dimension [73, 29, 21, 111], or express correlations between outputs in multi-task settings [55, 6, 2]. Moreover, the RBF and ARD kernels – arguably the

[1] In practice, $k$ depends on the conditioning of $K$, but is independent of $n$.

most popular kernels in use – decompose into product kernels.

In this chapter, we will address both of these limitations of iterative methods – improving modularity while simultaneously alleviating the curse of dimensionality. This is possible by once again exploiting structure inherent in the covariance function: our approach to alleviating the exponential dependence on dimension *crucially* relies on the fact that many popular kernels decompose as a product of one dimensional covariance functions. We will exploit this inherent product structure for fast MVMs. In particular:

1. We demonstrate that MVMs with product kernels can be approximated efficiently by computing the Lanczos decomposition of each component kernel. If MVMs with a kernel $K$ can be performed in $\mathcal{O}\left(\mu(K)\right)$ time, then MVMs with the element-wise product of $d$ kernels can be approximated in $\mathcal{O}\left(dk\mu(K) + k^3 n \log d\right)$ time, where $k$ is typically a very small constant.

2. Our fast product-kernel MVM algorithm, entitled *SKIP*, enables the use of structured kernel interpolation with product kernels without resorting to the exponential complexity of Kronecker products. SKIP can be applied even when the product kernels use different interpolation grids, and enables GP inference and learning in $\mathcal{O}\left(dn + dm \log m\right)$ for products of $d$ kernels.

3. We apply SKIP to high-dimensional regression problems by expressing $d$-dimensional kernels as the product of $d$ one-dimensional kernels. Despite this expression being exact for many popular kernels, this formulation affords an *exponential improvement* over the standard SKI complexity of $\mathcal{O}\left(n + dm^d \log m\right)$, and achieving state of the art performance over popular inducing point methods [36, 97].

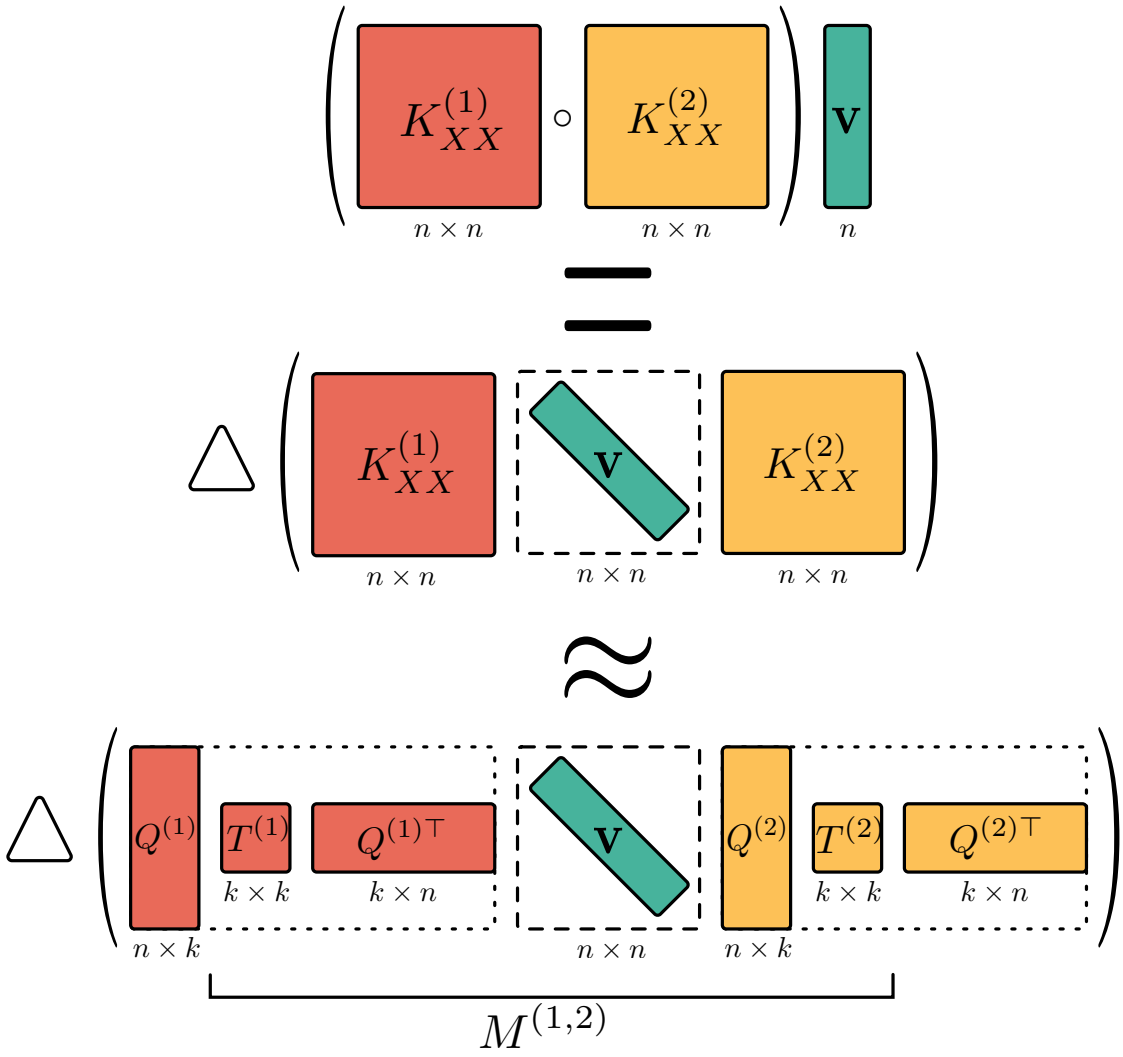


Figure 6.1: Computing fast matrix-vector multiplies (MVMs) with the product kernel $K_{XX}^{(1)} \circ K_{XX}^{(2)}$. **1:** Rewrite the element-wise product as the diagonal $\Delta(\cdot)$ of a product of matrices. **2:** Compute the rank-$k$ Lanczos decomposition of $K_{XX}^{(1)}$ and $K_{XX}^{(2)}$.

4. We demonstrate that SKIP can reduce the complexity of multi-task GPs (MT-GPs) to $\mathcal{O}\left(n + m \log m + s\right)$ for a problem with $s$ tasks. We exploit this fast inference, developing a model that discovers cluster of tasks using Gibbs sampling.

## 6.2 Matrix-vector multiplication with product kernels

Suppose a kernel separates as a product as follows:

$$k(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{d} k^{(i)}(\mathbf{x}, \mathbf{x}'). \tag{6.1}$$

Given a training data set $\mathbf{X} = [\mathbf{x}_1, \ldots \mathbf{x}_n]$, the kernel matrix $K$ resulting from the product of kernels in (6.1) can be expressed as $K = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(d)}$, where $\circ$ represents element-wise multiplication. In other words:

$$\left[K_{XX}^{(1)} \circ K_{XX}^{(2)}\right]_{ij} = \left[K_{XX}^{(1)}\right]_{ij} \left[K_{XX}^{(2)}\right]_{ij}. \tag{6.2}$$

The key limitation we must deal with is that, unlike a sum of matrices, vector multiplication does not distribute over the elementwise product:

$$\left(K^{(1)} \circ K^{(2)}\right) \mathbf{v} \neq \left(K^{(1)}\mathbf{v}\right) \circ \left(K^{(2)}\mathbf{v}\right). \tag{6.3}$$

We will assume we have access to fast MVMs for each component kernel matrix $K^{(i)}$. Without fast MVMs, there is a trivial solution to computing the elementwise matrix vector product: explicitly compute the kernel matrix $K$ in $\mathcal{O}\left(dn^2\right)$ time and then compute $K\mathbf{v}$. We further assume that $K^{(i)}$ admits a low rank approximation, following prior work on inducing point methods [89, 97, 113, 36].

**A naive algorithm for a two-kernel product.** We initially assume for simplicity that there are only $d = 2$ components kernels in the product. We will then show how to extend the two kernel case to arbitrarily sized product kernels. We seek to perform matrix vector multiplies:

$$(K_{XX}^{(1)} \circ K_{XX}^{(2)})\mathbf{v} \tag{6.4}$$

Eq. (6.4) may be expressed in terms of matrix-matrix multiplication using the following identity:

$$K\mathbf{v} = (K_{XX}^{(1)} \circ K_{XX}^{(2)})\mathbf{v} = \Delta(K_{XX}^{(1)}\, D_{\mathbf{v}}\, K_{XX}^{(2)\top}), \tag{6.5}$$

where $D_{\mathbf{v}}$ is a diagonal matrix whose elements are $\mathbf{v}$ (6.1), and $\Delta(M)$ denotes the diagonal of $M$. Because $D_{\mathbf{v}}$ is an $n \times n$ matrix, computing the entries of $K\mathbf{v}$ naively requires $n$ matrix-vector multiplies with $K_{XX}^{(1)}$ and $K_{XX}^{(2)}$. The time complexity to compute (6.5) is therefore $\mathcal{O}\left(n\mu(K_{XX}^{(1)}) + n\mu(K_{XX}^{(2)})\right)$. This reformulation does not naively offer any time savings.

**Exploiting low-rank structure.** Suppose however that we have access to rank-$k$ approximations of $K_{XX}^{(1)}$ and $K_{XX}^{(2)}$:

$$K_{XX}^{(1)} \approx Q^{(1)}T^{(1)}Q^{(1)\top}, \quad K_{XX}^{(2)} \approx Q^{(2)}T^{(2)}Q^{(2)\top},$$

where $Q^{(1)}$, $Q^{(2)}$ are $n \times k$ and $T^{(1)}$, $T^{(2)}$ are $k \times k$ (6.1). This rank decomposition makes the MVM significantly cheaper to compute. Plugging these decompositions in to (6.5), we derive:

$$K\mathbf{v} = \Delta\left(Q^{(1)}T^{(1)}Q^{(1)\top}\, D_{\mathbf{v}}\, Q^{(2)}T^{(2)}Q^{(2)\top}\right). \tag{6.6}$$

We prove the following key lemma in about (6.6) in Section 6.3:

**Lemma 6.2.1.** *Suppose that $K_{XX}^{(1)} = Q^{(1)}T^{(1)}Q^{(1)\top}$ and $K_{XX}^{(2)} = Q^{(1)}T^{(1)}Q^{(1)\top}$, where $Q^{(1)}$ and $Q^{(2)}$ are $n \times k$ matrices and $T^{(1)}$ and $T^{(2)}$ are $k \times k$. Then $(K_{XX}^{(1)} \circ K_{XX}^{(2)})\mathbf{v}$ can be computed with* (6.6) *in $\mathcal{O}\left(k^2 n\right)$ time.*

Therefore, if we can efficiently compute low-rank decompositions of $K^{(1)}$ and $K^{(2)}$, then we immediately apply 6.2.1 to perform fast MVMs.

**Computing low-rank structure.** With 6.2.1, we have reduced the problem of computing MVMs with $K$ to that of constructing low-rank decompositions for $K_{XX}^{(1)}$ and $K_{XX}^{(2)}$. Since we are assuming we can take fast MVMs with these kernel matrices, we now turn to the *Lanczos decomposition* [53, 68]. The Lanczos decomposition is an iterative algorithm that takes a symmetric matrix $A$ and probe vector $b$ and returns $Q$ and $T$ such that $A = QTQ^\top$, with $Q$ orthogonal.

This decomposition is exact after $n$ iterations, i.e. $A = QTQ^\top$. However, suppose we were to only compute $k < n$ columns of $Q$. Then, $Q_k T_k Q_k^\top$ is an effective low-rank approximation of $A$ [66, 86]. Unlike standard low rank approximations (such as the singular value decomposition), the algorithm for computing the Lanczos decomposition $K_{XX}^{(i)} = Q^{(i)}T^{(i)}Q^{(i)\top}$ requires only $k$ MVMs, leading to the following lemma:

**Lemma 6.2.2.** *Suppose that MVMs with $K_{XX}^{(i)}$ can be computed in $\mathcal{O}\left(\mu(K_{XX}^{(i)})\right)$ time. Then the rank-$k$ Lanczos decomposition $K_{XX}^{(i)} \approx Q_k^{(i)} T_k^{(i)} Q_k^{(i)\top}$ can be computed in $\mathcal{O}\left(k\mu(K_{XX}^{(i)})\right)$ time.*

The above discussion motivates the following algorithm for computing $(K_{XX}^{(1)} \cdot K_{XX}^{(2)})\mathbf{v}$, which is summarized by 6.1: First, compute the rank-$k$ Lanczos decomposition of each matrix; then, apply (6.6). Lemmas 6.2.2 and 6.2.1 together imply that this takes $\mathcal{O}\left(k\mu(K_{XX}^{(1)}) + k\mu(K_{XX}^{(2)}) + k^2 n\right)$ time.

**Extending to product kernels with three components.** Now consider a kernel that decomposes as the product of three components, $k(\mathbf{x}, \mathbf{x}') = k^{(1)}(\mathbf{x}, \mathbf{x}')k^{(2)}(\mathbf{x}, \mathbf{x}')k^{(3)}(\mathbf{x}, \mathbf{x}')$. An MVM with this kernel is given by $K\mathbf{v} = (K_{XX}^{(1)} \circ K_{XX}^{(2)} \circ K_{XX}^{(3)})\mathbf{v}$. Define $\tilde{K}_{XX}^{(1)} = K_{XX}^{(1)} \circ K_{XX}^{(2)}$ and $\tilde{K}_{XX}^{(2)} = K_{XX}^{(3)}$. Then

$$K\mathbf{v} = (\tilde{K}_{XX}^{(1)} \circ \tilde{K}_{XX}^{(2)})\mathbf{v}\,, \tag{6.7}$$

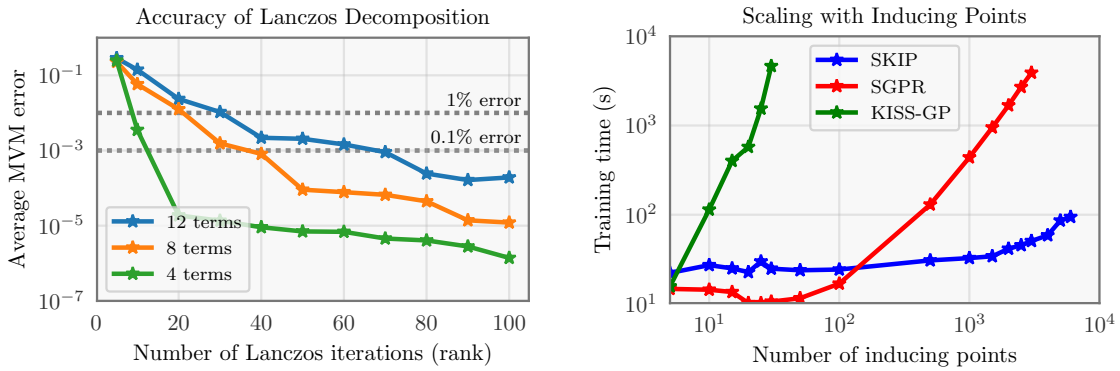


Figure 6.2: **Left:** Relative error of MVMs computed using SKIP compared to the exact value $K\mathbf{v}$. **Right:** Training time as a function of the number of inducing points *per dimension* on the Power dataset. KISS-GP scales badly here, because the required *total* number of inducing points scales exponentially with the number of dimensions. On the power dataset $d = 4$.

reducing the three component problem back to two components. To compute the Lanczos decomposition of $\tilde{K}_{XX}^{(1)}$, we use the method described above for computing MVMs with $K_{XX}^{(1)} \circ K_{XX}^{(2)}$.

**Extending to product kernels with many components.** The approach for the three component setting leads naturally to a divide and conquer strategy. Given a kernel matrix $K = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(d)}$ we define

$$\tilde{K}_{XX}^{(1)} = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(\frac{d}{2})} \tag{6.8}$$

$$\tilde{K}_{XX}^{(2)} = K_{XX}^{(\frac{d}{2}+1)} \circ \cdots \circ K_{XX}^{(d)}, \tag{6.9}$$

which lets us rewrite $K = \tilde{K}_{XX}^{(1)} \circ \tilde{K}_{XX}^{(2)}$. By applying this splitting recursively, we can compute matrix-vector multiplies with $K$, leading to the following running time complexity, which we prove in Section 6.3:

**Theorem 6.2.3.** *Suppose that $K = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(d)}$, and that computing a matrix-vector multiply with any $K_{XX}^{(i)}$ requires $\mathcal{O}\left(\mu(K_{XX}^{(1)})\right)$ operations. Computing an MVM with $K$ requires $\mathcal{O}\left(dk\mu(K^{(i)}) + k^3 n \log d + k^2 n\right)$ time, where $k$ is the rank of the Lanczos decomposition used.*

**Sequential MVMs.** If we are computing many MVMs with the same matrix, then we can further reduce this complexity by caching the Lanczos decomposition. The terms $\mathcal{O}\left(dk\mu(K_{XX}^{(i)}) + k^3 n \log d\right)$ represent the time to construct the Lanczos decomposition. However, note that this decomposition is not dependent on the vector that we wish to multiply with. Therefore, if we save the decomposition for future computation, we have the following corollary:

**Corollary 6.2.4.** *Any subsequent MVMs with $K$ require $\mathcal{O}\left(k^2 n\right)$ time.*

If matrix-vector multiplications with $K_{XX}^{(i)}$ can be performed with significantly fewer than $n^2$ operations, this results in a significant complexity improvement over explicitly computing the full kernel matrix $K$.

### 6.2.1 Structured kernel interpolation for products (SKIP)

So far we have supposed access to fast MVMs with each constituent kernel matrix of an elementwise (Hadamard) product: $K = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(d)}$. To achieve fast MVMs, we apply the SKI approximation (Section 2.5) to each component as:

$$K_{XX}^{(i)} = W^{(i)} K_{UU} W^{(i)\top}. \tag{6.10}$$

When using SKI approximations, the running time of our product kernel inference technique with $p$ iterations of CG becomes $\mathcal{O}\left(dk(n + m \log m) + k^3 n \log d + pk^2 n\right)$. The running time of SKIP is compared to that of other inference techniques in 6.1.

## 6.3 Proofs for key results

In this section, we briefly provide proofs for the two key results above, Lemma 6.2.1 and Theorem 6.2.3.

### 6.3.1 Proof of Lemma 6.2.1

Letting $\mathbf{q}_i^{(1)}$ denote the $i^{th}$ row of $Q^{(1)}$ and $\mathbf{q}_i^{(2)}$ denote the $i^{th}$ row of $Q^{(2)}$, we can express the $i^{th}$ entry $K\mathbf{v}$, $[K\mathbf{v}]_i$ as:

$$[K\mathbf{v}]_i = \mathbf{q}_i^{(1)} T^{(1)} Q^{(1)\top} \, D_{\mathbf{v}} \, Q^{(2)} T^{(2)} \mathbf{q}_i^{(2)\top}$$

To evaluate this for all $i$, we first once compute the $k \times k$ matrix:

$$M^{(1,2)} = T^{(1)} Q^{(1)\top} \, D_{\mathbf{v}} \, Q^{(2)} T^{(2)}.$$

This can be done in $O(nk^2)$ time. $T^{(1)}Q^{(1)\top}$ and $Q^{(2)}T^{(2)}$ can each be computed in $O(nk^2)$ time, as the $Q$ matrices are $n \times k$ and the $T$ matrices are $n \times k$. Multiplying one of the results by $D_{\mathbf{v}}$ takes $\mathcal{O}(nk)$ time as it is diagonal. Finally, multiplying the two resulting $n \times k$ matrices together takes $\mathcal{O}(nk^2)$ time.

After computing $M^{(1,2)}$, we can compute each element of the matrix-vector multiply as:

$$[K\mathbf{v}]_i = \mathbf{q}_i^{(1)} M^{(1,2)} \mathbf{q}_i^{(2)\top}.$$

Because $M^{(1,2)}$ is $k \times k$, each of these takes $\mathcal{O}(k)$ time to compute. Since there are $n$ entries to evaluate in the MVM $K\mathbf{v}$ in total, the total time requirement after computing $M^{(1,2)}$ is $\mathcal{O}(kn)$ time. Thus, given low rank structure, we can compute $K\mathbf{v}$ in $\mathcal{O}(k^2 n)$ time total.

### 6.3.2 Proof of Theorem 6.2.3

Given the Lanczos decompositions of $\tilde{K}^{(1)} = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(a)}$ and $\tilde{K}^{(2)} = K_{XX}^{(a+1)} \circ \cdots \circ K_{XX}^{(d)}$, we can compute matrix-vector multiplies with $\tilde{K}^{(1)} \circ \tilde{K}^{(2)}$ in $\mathcal{O}\left(k^2 n\right)$ time each. This lets us compute the Lanczos decomposition of $\tilde{K}^{(1)} \circ \tilde{K}^{(2)}$ in $\mathcal{O}\left(k^3 n\right)$ time.

For clarity, suppose first that $d = 3$, i.e., $K = K_{XX}^{(1)} \circ K_{XX}^{(2)} \circ K_{XX}^{(3)}$. We first Lanczos decompose $K_{XX}^{(1)}$, $K_{XX}^{(2)}$ and $K_{XX}^{(3)}$. Assuming for simplicty that MVMs with each matrix take the same amount of time, This takes $\mathcal{O}\left(k\mu(K_{XX}^{(i)})\right)$ time total. We then use these Lanczos decompositions to compute matrix-vector multiples with $\tilde{K}_{XX}^{(1)}$ in $\mathcal{O}\left(k^2 n\right)$time each. This allows us to Lanczos decompose it in $\mathcal{O}\left(k^3 n\right)$ time total. We can then compute matrix-vector multiplications $K\mathbf{v}$ in $\mathcal{O}\left(k^2 n\right)$ time.

In the most general setting where $K = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(d)}$, we first Lanczos decompose the $d$ component matrices in $\mathcal{O}\left(dk\mu(K^{(i)})\right)$ and then perform $\mathcal{O}\left(\log d\right)$ merges as described above, each of which takes $\mathcal{O}\left(k^3 n\right)$ time. After computing all necessary Lanczos decompositions, matrix-vector multiplications with $K$ can be performed in $\mathcal{O}\left(k^2 n\right)$ time.

As a result, a single matrix-vector multiply with $K$ takes $\mathcal{O}\left(dk\mu(K^{(i)}) + k^3 n \log d + k^2 n\right)$ time. With the Lanczos decompositions precomputed, multiple MVMs in a row can be performed significantly faster. For example, running $p$ iterations of conjugate gradients with $K$ takes $\mathcal{O}\left(dk\mu(K^{(i)}) + k^3 n \log d + pk^2 n\right)$ time.

Table 6.1: Asymptotic complexities of a single inference step with $n$ training examples and $m$ inducing points, using $k$ Lanczos iterations for SKIP and $p$ CG iterations for SKIP and KISS-GP.

| **Method** | **Complexity of 1 Inference Step** |
|---|---|
| GP (Chol) | $\mathcal{O}\left(n^3\right)$ |
| GP (MVM) | $\mathcal{O}\left(pn^2\right)$ |
| SVGP | $\mathcal{O}\left(nm^2 + m^3 + dnm\right)$ |
| KISS-GP | $\mathcal{O}\left(pn + pdm^d \log m\right)$ |
| SKIP | $\mathcal{O}\left(dkn + dkm \log m + k^3 n \log d + pk^2 n\right)$ |

## 6.4 Evaluating MVM accuracy

We first evaluate the accuracy of our proposed approach with product kernels in a controlled synthetic setting. We draw $2500$ data points in $d$ dimensions from $\mathcal{N}(0, I)$ and compute an RBF kernel matrix with lengthscale $1$ over these data points. We evaluate the relative error of SKIP compared to exact MVMs as a function of $k$ – the number of Lanczos iterators. We perform this test for 4, 8, and 12 dimensional data, resulting in a product kernel with 4, 8, and 12 components respectively. The results, averaged over 100 trials, are shown in 6.2 (left). Even in the 12 dimensional setting, an extremely small value of $k$ is sufficient to get very accurate MVMs: less than 1% error is achieved when $k = 30$. For a discussion of increasing error with dimensionality, see 6.6. In future experiments, we set the maximum number of Lanczos iterations to $100$, but note that the convergence criteria is typically met far sooner.

## 6.5 An exponential improvement to SKI

[113] use a Kronecker decomposition of $K_{UU}$ to apply SKI for $d > 1$ dimensions, which requires a fully connected multi-dimensional grid of inducing points $U$. Thus if we wish to have $m$ distinct inducing point values for each dimension, the grid requires $m^d$ inducing points – i.e. MVMs with the SKI approximate $K_{XX}$ require $\mathcal{O}\left(n + dm^d \log m\right)$ time. It is therefore computationally infeasible to apply SKI in more than about five dimensions.

However, using the proposed SKIP method of Section 6.2, we can reduce the running time complexity of SKI in $d$ dimensions from exponential $\mathcal{O}\left(n + dm^d \log m\right)$ to linear $\mathcal{O}\left(dn + dm \log m\right)$! If we express a $d$-dimensional kernel as the product of $d$ one-dimensional kernels, then each component kernel requires only $m$ grid points, rather than $m^d$. For the RBF and ARD kernels, decomposing the kernel in this way yields the same kernel function.

**Datasets.** We evaluate SKIP on six benchmark datasets. The precipitation dataset contains hourly rainfall measurements from hundreds of stations around the country. The remaining datasets are taken from the UCI machine learning dataset repository. KISS-GP is not applicable when $d > 5$, and the full GP is not applicable on the four largest datasets.

**Methods.** We compare against the popular sparse variational Gaussian processes (SGPR) [97, 36] implemented in GPflow [60]. We also compare to our GPU implementation of KISS-GP where possible, as well as our GPU implementation of the full GP on the two smallest datasets. All experiments were run on an NVIDIA Titan Xp. We evaluate SGPR using 200, 400 and 800 inducing points.

Table 6.2: Comparison of SKIP and other methods on higher dimensional datasets. In this table, $m$ is the *total* number of inducing points, rather than number of inducing points per dimension. (*We use $m = 100$ for SKIP on all datasets except precipitation, where we use $m = 120$K.)

| Dataset | Metric | Full GP | SGPR ($m = 200$) | SGPR ($m = 400$) | SGPR ($m = 800$) | KISS-GP ($m = 120$K) | SKIP ($m = 100$)* |
|---|---|---|---|---|---|---|---|
| **Pumadyn** ($n = 8192$, $d = 32$) | **Test MAE** | **0.721** | 0.766 | 0.766 | 0.766 | – | 0.766 |
| | **Train Time ($s$)** | **4** | 28 | 67 | 235 | – | 65 |
| **Elevators** ($n = 16599$, $d = 18$) | **Test MAE** | **0.072** | 0.157 | 0.157 | 0.157 | – | **0.072** |
| | **Train Time ($s$)** | **12** | 46 | 122 | 425 | – | 23 |
| **Precipitation** ($n = 628474$, $d = 3$) | **Test MAE** | – | 14.79 | – | – | **9.81** | 14.08 |
| | **Train Time ($s$)** | – | 1432 | – | – | 615 | **34.16** |
| **KEGG** ($n = 48827$, $d = 22$) | **Test MAE** | – | 0.101 | 0.093 | 0.087 | – | **0.065** |
| | **Train Time ($s$)** | – | 116 | 299 | 9926 | – | **66** |
| **Protein** ($n = 45730$, $d = 9$) | **Test MAE** | – | 7.219 | 4.97 | 4.72 | – | **1.97** |
| | **Train Time ($s$)** | – | 139 | 397 | 1296 | – | **35** |
| **Video** ($n = 68784$, $d = 16$) | **Test MAE** | – | 6.836 | 6.463 | 6.270 | – | **5.621** |
| | **Train Time ($s$)** | – | 113 | 334 | 1125 | – | **57** |

All models use the RBF kernel and a constant prior mean function. We optimize hyperparameters with ADAM using default optimization parameters.

**Discussion.** The results of our experiments are shown in Table 6.2. On the two smallest datasets, the Full GP model outperforms all other methods in terms of speed. This is due to the overhead added by inducing point methods significantly outweighing simple solves with conjugate gradients with such little data. SKIP is able to match the error of the full GP model on elevators, and all methods have comparable error on the Pumadyn dataset.

On the precipitation dataset, inference with standard KISS-GP is still tractable due to the low dimensionality, and KISS-GP is both fast and accurate. Using SKIP results in higher error than KISS-GP, because we were able to use significantly fewer Lanczos iterations for our approximate MVMs than on other datasets due to the space complexity. We discuss the space complexity limitation further in the discussion section. Nevertheless, SKIP still performs better than SGPR. SGPR results with 400 and 800 inducing points are unavailable due to GPU memory

constraints.

On the remaining datasets, SKIP is able to achieve comparable or better overall error than SGPR, but with a significantly lower runtime. There are two important things to note from these results. First, because SKIP interpolates each dimension independently, far fewer inducing points are needed overall: SKIP does not need to cover the full $d$ dimensional space with inducing points. Second, while SKIP significantly improves on the curse of dimensionality suffered by SKI, the running time of SKIP still depends linearly on the dimensionality, a fact we discuss further later in the chapter. We notice that, while SKIP still performs better on the highest dimensional dataset (KEGG), its running time is significantly larger than for the other datasets.

## 6.6 Discussion

Beyond the exponential improvement to the running time complexity of SKI and the extension of MVM-based inference methods to product kernels, this chapter also raises the following foundational question about scalable GP inference: *given the ability to compute $A\mathbf{v}$ and $B\mathbf{v}$ quickly for matrices $A$ and $B$, how do we compute $(A \circ B)\mathbf{v}$ quickly?* We have shown an answer to this question can dramatically improve the scalability and general applicability of MVM-based methods for fast Gaussian processes. In particular, we have introduced a general technique for fast MVMs exploiting product kernel structure, enabling SKI inference that scales linearly instead of exponentially with dimension. We conclude the chapter with a discussion of some practical limitations and related work.

**Other inducing point methods.** A variety of inducing point methods other than SKI exist [89, 97]. Like SKI, many of these methods involve approximating the kernel using a set of inducing points. For example, the popular subset of regressors (SoR) method (used, for example, in the SGPR method we compare to above) approximates $K_{XX'}^{\text{SoR}} = K_{XU}K_{UU}^{-1}K_{UX'}$. These methods can be efficiently implemented using iterative MVM strategies: calculating $[K_{XU}K_{UU}^{-1}K_{UX'}]\mathbf{v}$ takes $\mathcal{O}\left(nm + m^2\right)$ time after initial work to decompose $K_{UU}$. However, applying SKIP as an improvement over SKI results in both the most dramatic improvement in running time (from exponential in $d$ to linear), and also results in the fastest overall running time complexity we are aware of for $d$ dimensional Gaussian process regression.

Recent work extends inducing point methods to be compatible with stochastic gradient methods [36, 37]. As demonstrated in [110], these stochastic variational methods are fully compatible with the iterative MVM approach and SKI, and we anticipate these approaches can be used to further accelerate SKIP as well.

**Stochastic diagonal estimation and small lengthscales.** Our method relies primarily on quickly computing the diagonal in Equation (6.5). Techniques exist for stochastic diagonal estimation [23, 42, 82, 5]. We found that these methods converge slower than the Lanczos approach described here. However, the Lanczos decomposition may provide a poor approximation in cases where the kernel matrix is high rank, for example with extremely small lengthscales or nonsmooth kernels. Thus, stochastic diagonal estimation may be more appropriate for evaluating (6.5) in some cases.

**Higher-order product kernels.** A fundamental property of the Hadamard product is that rank$(A \circ B) \leq$ rank$(A)$rank$(B)$ suggesting that if we apply the Lanczos decomposition to the product of a large number of matrices, we may need more iterations to derive accurate approximations. In the limit, the SKI approximation $WK_{UU}W^\top$ can be used in place of the Lanczos decomposition in equation (6.5), resulting in an exact algorithm with $\mathcal{O}\left(dnm + dm^2 \log m\right)$ runtime. In practice this adaptation is rarely necessary, as the increasing lengthscale with dimensionality counteracts the rank increase of $K$.

**Space complexity.** To perform the matrix-vector multiplication algorithm described above, we must store the Lanczos decomposition of each component kernel matrix and intermediate matrices in the merge step for $\mathcal{O}\left(dkn\right)$ storage. This is better storage than the $\mathcal{O}\left(n^2\right)$ storage required in full GP regression, or $\mathcal{O}\left(nm\right)$ storage for standard inducing point methods, but worse than the linear storage requirements of SKI.

# CHAPTER 7

# DISCUSSION AND FUTURE DIRECTIONS

The goal of this thesis has been to explore the application of covariance structure to modeling functions, especially in the limited data regime. In particular, selecting the right covariance function for the task (whether manually or actively) can dramatically reduce sample complexity. Exploiting covariance structure allows for audiometric testing in a fraction of the time required by the standard test, and can lead to *exponentially* lower sample complexity for Bayesian optimization. By exploiting the product structure inherent in the RBF kernel, we can achieve *exponential* speed-ups for training Gaussian processes. This last result is a particularly compelling use of covariance function structure, as it leads to the fastest asymptotic complexity for Gaussian process training we are aware of.

**Large-scale Learning.** This thesis primarily deals with *restrictive* covariance functions: kernels that impose significant structure on the model and significantly reducing the size of the hypothesis class modeled by the Gaussian process. This is useful or even mandatory when dealing with very small amounts of data. The audiometric testing setting and the Bayesian optimization setting necessarily fit this setting, as the cost of acquiring each training example is assumed to be very high.

Alongside this thesis, an alternative approach to covariance function selection is *kernel learning*, in which a highly flexible (i.e., highly parameterized) covariance function is used. Recent examples of this include the spectral mixture kernel [108] and deep kernel learning [112, 110]. Unlike the types of structure largely dealt with in this thesis, these techniques greatly expand the flexiblity of

Gaussian processes, and can be used to absorb large amounts of data.

Because most of the applications discussed in this thesis fall in the small data regime, scalability has not been an issue, and we have used standard Gaussian process inference throughout. However, to take full advantage of kernel learning, inference and predictions with massive datasets must be made tractable. SKIP takes a significant step in this direction for product kernels, reducing the running time to linear in both the number of training examples $n$ (as was the case with standard SKI) and the number of features $d$ (instead of exponential). The predictive mean for a single test point can be computed in constant time with SKI, but computing predictive variances efficiently remains an open problem.

As present and future work address these scalability concerns and bring the time costs for training and predictions using Gaussian processes in line with other modern machine learning techniques, they become an attractive choice for probabilistic modeling. In low data settings such as active learning and Bayesian optimization, GPs have found significant applicability in large part due to the availabilty of model uncertainty estimates that allow for careful decision-making. If the running time cost of using GPs becomes negligible, this aspect of GP regression could prove to be an immensely useful augmentation of existing deep learning architectures by allowing for calibrated model uncertainty estimates without sacrificing accuracy or speed.

**Small-scale learning.** Despite the promise of large scale learning and the growing availability of large datasets, the ability to learn in the presence of very small datasets will quite likely to continue to be important. Bayesian optimization and the active learning setting discussed in our audiometric test are examples of set-

tings where learning must start by definition with no data at all, as each instance deals with an entirely new objective function or patient. While work exists that attempts to use multi-task Gaussian processes to transfer knowledge from one optimization task to another (e.g, [94]), this approach is largely applicable only to optimization problems within the same domain: in other words, it would be difficult to transfer knowledge from hyperparameter tuning to experimental design.

In these settings, highly parameterized models may not be as useful, as it can be difficult to generalize a very flexible model with only a handful of datasets. As a result, careful thought about more restrictive and informative priors can be greatly advantageous. It has largely been common practice with Bayesian optimization to assume that, because the function is unknown, a general purpose covariance function must be used. A central theme of this thesis is that covariance functions can be selected *actively*. While the function is unknown at first, after 20, 40, or 100 iterations of Bayesian optimization or active learning we may know a great deal about the function: enough to choose a better model. By intelligently discovering and exploiting this structure–even in the limited data setting–we can construct models that are *exponentially better* than otherwise possible.

## BIBLIOGRAPHY


[1] A. Ali, R. Caruana, and A. Kapoor. Active Learning with Model Selection. In *AAAI*, 2014.

[2] Mauricio A Álvarez and Neil D Lawrence. Computationally efficient convolved multiple output Gaussian processes. *Journal of Machine Learning Research*, 12(May):1459–1500, 2011.

[3] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

[4] Thomas C Bailey, Yixin Chen, Yi Mao, Chenyang Lu, Gregory Hackmann, Scott T Micek, Kevin M Heard, Kelly M Faulkner, and Marin H Kollef. A Trial of a Real-Time Alert for Clinical Deterioration in Patients Hospitalized on General Medical Wards. *Journal of Hospital Medicine*, 8(5):236–242, 2013.

[5] Costas Bekas, Effrosyni Kokiopoulou, and Yousef Saad. An estimator for the diagonal of a matrix. *Applied numerical mathematics*, 57(11):1214–1229, 2007.

[6] Edwin V Bonilla, Kian M Chai, and Christopher Williams. Multi-task Gaussian process prediction. In *NIPS*, pages 153–160, 2008.

[7] E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[8] Adam D Bull. Convergence rates of efficient global optimization algorithms. *The Journal of Machine Learning Research*, 12:2879–2904, 2011.

[9] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[10] Roberto Calandra, Jan Peters, Andre Seyfarth, and Marc P Deisenroth. An experimental evaluation of bayesian optimization on bipedal locomotion. In *International Conference on Robotics and Automation*, 2014.

[11] Raymond Carhart and James Jerger. Preferred Method for Clinical Determination of Pure-Tone Thresholds. *Journal of Speech and Hearing Disorders*, 24(4):330–345, 1959.

[12] Shane Carr, Roman Garnett, and Cynthia Lo. BASC: Applying Bayesian Optimization to the Search for Global Minima on Potential Energy Surfaces. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 898–907, 2016.

[13] Matt Carter and Jennifer C Shieh. *Guide to research techniques in neuroscience*. Academic Press, 2009.

[14] John P Cunningham, Krishna V Shenoy, and Maneesh Sahani. Fast gaussian process methods for point process intensity estimation. In *Proceedings of the 25th international conference on Machine learning*, pages 192–199. ACM, 2008.

[15] Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215. ACM, 2008.

[16] Manuel Don, Jos J Eggermont, and Derald E Brackmann. Reconstruction of the audiogram using brain stem responses and high-pass noise masking. *Annals of Otology, Rhinology and Laryngology.*, 88(3 Part 2, Supplement 57):1–20, 1979.

[17] Kun Dong, David Eriksson, Hannes Nickisch, David Bindel, and Andrew Gordon Wilson. Scalable log determinants for gaussian process kernel learning. In *NIPS*, 2017.

[18] David Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, Computational and Biological Learning Laboratory, University of Cambridge, 2014.

[19] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1166–1174, 2013.

[20] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure Discovery in Nonparametric Regression through Compositional Kernel Search. In *ICML*, pages 1166–1174, 2013.

[21] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1166–1174, 2013.

[22] David K Duvenaud, Hannes Nickisch, and Carl E Rasmussen. Additive gaussian processes. In *NIPS*, 2011.

[23] Jack K Fitzsimons, Michael A Osborne, Stephen J Roberts, and Joseph F Fitzsimons. Improved stochastic trace estimation using mutually unbiased bases. *arXiv preprint arXiv:1608.00117*, 2016.

[24] Jacob Gardner, Gustavo Malkomes, Roman Garnett, Kilian Q Weinberger, Dennis Barbour, and John P Cunningham. Bayesian active model selection with an application to automated audiometry. In *Advances in Neural Information Processing Systems*, pages 2377–2385, 2015.

[25] Jacob R Gardner, Xinyu Song, Kilian Q Weinberger, Dennis L Barbour, and John P Cunningham. Psychophysical detection testing with bayesian active learning. In *UAI*, pages 286–295, 2015.

[26] Roman Garnett, Michael A Osborne, and Philipp Hennig. Active Learning of Linear Embeddings for Gaussian Processes. In *UAI*, pages 230–239, 2014.

[27] Roman Garnett, Michael A Osborne, and Stephen J Roberts. Bayesian optimization for sensor set selection. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 209–219. ACM, 2010.

[28] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[29] Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of machine learning research*, 12(Jul):2211–2268, 2011.

[30] Rudolph E Gosztonyi Jr, Lawrence A Vassallo, and Joseph Sataloff. Audiometric reliability in industry. *Archives of Environmental Health: An International Journal*, 22(1):113–118, 1971.

[31] GPy. GPy: A gaussian process framework in python, since 2012.

[32] David M Green. A maximum-likelihood method for estimating thresholds in a yes–no task. *The Journal of the Acoustical Society of America*, 93(4):2096–2105, 1993.

[33] Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. Near-optimal sensor placements in gaussian processes. In *ICML*, 2005.

[34] Yuhong Guo and Russell Greiner. Optimistic active-learning using mutual information. In *IJCAI*, volume 7, pages 823–829, 2007.

[35] Trevor J Hastie and Robert J Tibshirani. *Generalized additive models*, volume 43. CRC press, 1990.

[36] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

[37] James Hensman, Alexander G de G Matthews, and Zoubin Ghahramani. Scalable variational gaussian process classification. In *AISTATS*. JMLR, 2015.

[38] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *NIPS*, pages 918–926, 2014.

[39] Neil Houlsby, José M Hernández-Lobato, and Zoubin Ghahramani. Cold-start Active Learning with Robust Ordinal Matrix Factorization. In *ICML*, pages 766–774, 2014.

[40] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

[41] Waiter Hughson and Harold Westlake. Manual for program outline for rehabilitation of aural casualties both military and civilian. *Transactions of the American Academy of Ophthalmology and Otolaryngology*, 48(Supplement):1–15, 1944.

[42] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

[43] Tomoharu Iwata, Neil Houlsby, and Zoubin Ghahramani. Active learning

for interactive visualization. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 342–350, 2013.

[44] James Jerger. Bekesy audiometry in analysis of auditory disorders. *Journal of Speech, Language, and Hearing Research*, 3(3):275–287, 1960.

[45] Kirthevasan Kandasamy, Jeff Schneider, and Barnabas Poczos. High dimensional bayesian optimisation and bandits via additive models. In David Blei and Francis Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 295–304. JMLR Workshop and Conference Proceedings, 2015.

[46] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160, 1981.

[47] Igor Kononenko. Machine Learning for Medical Diagnosis: History, State of the Art and Perspective. *Artificial Intelligence in Medicine*, 23(1):89–109, 2001.

[48] Andreas Krause and Carlos Guestrin. Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In *ICML 24*, 2007.

[49] Jouni Kuha. AIC and BIC: Comparisons of Assumptions and Performance. *Sociological Methods and Research*, 33(2):188–229, 2004.

[50] Johannes Kulick, Robert Lieck, and Marc Toussaint. Active Learning of Hyperparameters: An Expected Cross Entropy Criterion for Active Model Selection. *CoRR*, abs/1409.7552, 2014.

[51] S. R. Kulkarni, S. K. Mitter, and J. N. Tsitsiklis. Active learning using arbitrary binary valued queries. *Machine Learning*, 11:23–35, 1993.

[52] Malte Kuss and Carl Edward Rasmussen. Assessing Approximate Inference for Binary Gaussian Process Classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

[53] Cornelius Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.

[54] Marjorie R Leek, Judy R Dubno, Ning-ji He, and Jayne B Ahlstrom. Experience with a yes–no single-interval maximum-likelihood procedure. *The Journal of the Acoustical Society of America*, 107(5):2674–2684, 2000.

[55] D. J. C. MacKay. Introduction to Gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, pages 133–165. Springer, Berlin, 1998.

[56] David JC MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[57] Ted Madison et al. Guidelines for manual pure-tone threshold audiometry. 2005.

[58] Nimalan Mahendran, Ziyu Wang, Firas Hamze, and Nando D Freitas. Adaptive mcmc with bayesian optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 751–760, 2012.

[59] Gustavo Malkomes, Chip Schaff, and Roman Garnett. Bayesian optimization for automated model selection. In *Advances in Neural Information Processing Systems 29*, 2016.

[60] Alexander G de G Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. Gpflow: A gaussian process library using tensorflow. *Journal of Machine Learning Research*, 18(40):1–6, 2017.

[61] D McBride and S Williams. Audiometric notch as a sign of noise induced hearing loss. *Occupational Environmental Medicine*, 58(1):46–51, 2001.

[62] Christian Meyer-Bisch. Audioscan: a high-definition audiometry technique based on constant-level frequency sweeps-a new method with new hearing indicators. *International Journal of Audiology*, 35(2):63–72, 1996.

[63] Thomas P Minka. Expectation Propagation for Approximate Bayesian Inference. In *UAI*, pages 362–369, 2001.

[64] J. Mockus, V. Tiesis, and A. Zilinskas. The application of bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

[65] Kevin P Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[66] Hannes Nickisch, Rolf Pohmann, Bernhard Schölkopf, and Matthias Seeger. Bayesian experimental design of magnetic resonance imaging sequences. In *NIPS*, pages 1441–1448, 2009.

[67] Özcan Özdamar, Rebecca E Eilers, Edward Miskiel, and Judith Widen. Classification of audiograms by sequential testing using a dynamic bayesian procedure. *The Journal of the Acoustical Society of America*, 88(5):2171–2179, 1990.

[68] Christopher C Paige. Computational variants of the lanczos method for the eigenproblem. *IMA Journal of Applied Mathematics*, 10(3):373–381, 1972.

[69] Alex Pentland. Maximum likelihood estimation: The best pest. *Attention, Perception, & Psychophysics*, 28(4):377–379, 1980.

[70] Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

[71] Adrian E Raftery. Approximate Bayes Factors and Accounting for Model Uncertainty in Generalised Linear Models. *Biometrika*, 83(2):251–266, 1996.

[72] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[73] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[74] C.E. Rasmussen and C.K.I. Williams. Gaussian processes for machine learning. MIT Press, 2006.

[75] DW Robinson. Long-term repeatability of the pure-tone hearing threshold and its relation to noise exposure. *British journal of audiology*, 25(4):219–235, 1991.

[76] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, pages 441–448, 2001.

[77] Yunus Saatçi. *Scalable inference for structured Gaussian process models*. PhD thesis, University of Cambridge, 2012.

[78] Suchi Saria, Anand K. Rajani, Jeffrey Gould, Daphne L. Koller, and Anna A Penn. Integration of Early Physiological Responses Predicts Later Illness Severity in Preterm Infants. *Science Translational Medicine*, 2(48):48ra65, 2010.

[79] U Schiefer, J Pätzold, and F Dannheim. Konventionelle perimetrie. *Der Ophthalmologe*, 102(6):627–646, 2005.

[80] Nicolas Schmuziger, Rudolf Probst, and Jacek Smurzynski. Test-retest reliability of pure-tone thresholds from 0.5 to 16 khz using sennheiser hda 200 and etymotic research er-2 earphones. *Ear and hearing*, 25(2):127–132, 2004.

[81] Gideon Schwarz. Estimating the Dimension of a Model. *Annals of Statistics*, 6(2):461–464, 1978.

[82] Marco Selig, Niels Oppermann, and Torsten A Enßlin. Improving stochastic estimates with inference methods: Calculating matrix diagonals. *Physical Review E*, 85(2):021134, 2012.

[83] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[84] Josef Shargorodsky, Sharon G Curhan, Gary C Curhan, and Roland Eavey. Change in Prevalence of Hearing Loss in US Adolescents. *Journal of the American Medical Association*, 304(7):772–778, 2010.

[85] Jonathan Richard Shewchuk et al. An introduction to the conjugate gradient method without the agonizing pain, 1994.

[86] Horst D Simon and Hongyuan Zha. Low-rank matrix approximation using the lanczos bidiagonalization process with applications. *SIAM Journal on Scientific Computing*, 21(6):2257–2274, 2000.

[87] Cas Smits, Theo S Kapteyn, and Tammo Houtgast. Development and validation of an automatic speech-in-noise screening test by telephone. *International journal of audiology*, 43(1):15–28, 2004.

[88] Alex J Smola and Bernhard Schölkopf. *Learning with kernels*. GMD-Forschungszentrum Informationstechnik, 1998.

[89] Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *NIPS*, pages 1257–1264, 2006.

[90] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *NIPS 2012*, pages 2960–2968. 2012.

[91] X. D. Song, B. M. Wallace, J. R. Gardner, N. M. Ledbetter, K. Q. Weinberger, and D. L Barbour. Fast, continuous audiogram estimation using machine learning. *Ear and Hearing*, 2015.

[92] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[93] De Wet Swanepoel, Hermanus C Myburgh, David M Howe, Faheema Mahomed, and Robert H Eikelboom. Smartphone hearing screening with integrated quality control and data management. *International journal of audiology*, 53(12):841–849, 2014.

[94] K. Swersky, J. Snoek, and R. P. Adams. Multi-task bayesian optimization. In *NIPS 2013*, pages 2004–2012. 2013.

[95] Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task Bayesian optimization. In *NIPS*, pages 2004–2012, 2013.

[96] MiM Taylor and C Douglas Creelman. Pest: Efficient estimates on probability functions. *The Journal of the Acoustical Society of America*, 41(4A):782–787, 1967.

[97] Michalis K Titsias. Variational learning of inducing variables in sparse gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 567–574, 2009.

[98] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.

[99] Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr (f (a)) via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017.

[100] Marcel SMG Vlaming, Robert C MacKinnon, Marije Jansen, and David R

Moore. Automated screening for high-frequency hearing loss. *Ear and hearing*, 35(6):667, 2014.

[101] Grace Wahba. *Spline models for observational data*. SIAM, 1990.

[102] Z. Wang, M. Zoghi, F. Hutter, D. Matheson, and N. de Freitas. Bayesian optimization in a billion dimensions via random embeddings. *arXiv preprint arXiv:1301.1942*, 2013.

[103] Zheng Wang, Ming-Jun Lai, Zhaosong Lu, Wei Fan, Hasan Davulcu, and Jieping Ye. Orthogonal rank-one matrix pursuit for low rank matrix completion. *SIAM Journal on Scientific Computing*, 37(1):A488–A514, 2015.

[104] Charles S Watson, Gary R Kidd, James D Miller, Cas Smits, and Larry E Humes. Telephone screening tests for functionally impaired hearing: Current use in seven countries and development of a us version. *Journal of the American Academy of Audiology*, 23(10):757–767, 2012.

[105] Herbert S. Wilf. *Generatingfunctionology*. A. K. Peters, Ltd., Natick, MA, USA, 2006.

[106] C. Williams and C. Rasmussen. Gaussian processes for regression. In *NIPS*, 1996.

[107] Victoria Williams-Sanchez, Rachel A McArdle, Richard H Wilson, Gary R Kidd, Charles S Watson, and Andrea L Bourne. Validation of a screening test of auditory function using the telephone. *Journal of the American Academy of Audiology*, 25(10):937–951, 2014.

[108] Andrew Wilson and Ryan Adams. Gaussian process kernels for pattern discovery and extrapolation. In *ICML*, pages 1067–1075, 2013.

[109] Andrew G Wilson, Elad Gilboa, Arye Nehorai, and John P Cunningham. Fast Kernel Learning for Multidimensional Pattern Extrapolation. In *NIPS*, pages 3626–3634, 2014.

[110] Andrew G Wilson, Zhiting Hu, Ruslan R Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594, 2016.

[111] Andrew Gordon Wilson. Covariance kernels for fast automatic pattern dis-

covery and extrapolation with gaussian processes. *University of Cambridge*, 2014.

[112] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

[113] Andrew Gordon Wilson and Hannes Nickisch. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *ICML*, pages 1775–1784, 2015.

[114] F Zhao, D Stephens, and C Meyer-Bisch. The audioscan: a high frequency resolution audiometric technique and its clinical applications. *Clinical Otolaryngology & Allied Sciences*, 27(1):4–10, 2002.

[115] Fei Zhao and Dafydd Stephens. Analyses of notches in audioscan and dpoaes in subjects with normal hearing. *International Journal of Audiology*, 37(6):335–343, 1998.